Radio Shack TRS-80™ Microcomputer NEWS

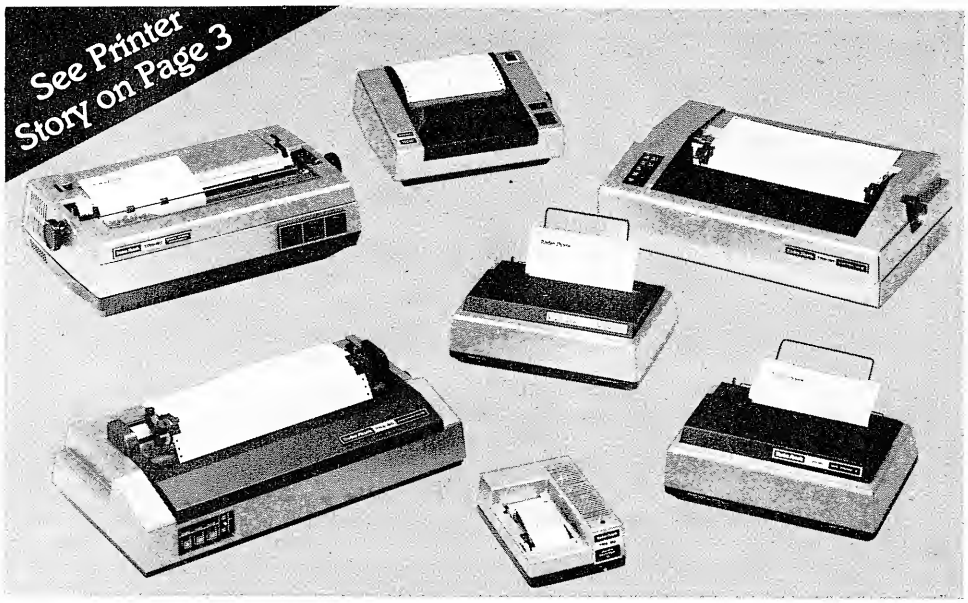
May, 1981

P.O. Box 2910, Fort Worth, Texas 76101

THE MICROCOMPUTER NEWSLETTER PUBLISHED FOR TRS-80 OWNERS

Volume 3, Issue 5

Newsletter Index
In this issue . . .



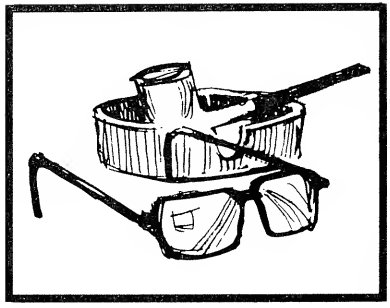
Copyright Law Amended

In December of 1980, Congress amended the United States Copyright Law to permit computer programs to be copyrighted. We recently received this information from Radio Shack's legal department, along with a short summary of the changes and copies of the amendments. We thought this information might be of interest to you.

In general we can summarize the changes by saying that the Copyright Law was amended to specifically state that a computer program was copyrightable. The amendment also provides that you, as the owner of a copy of a copyrighted program, are authorized to make copies of the program if you need to do so in order to use the program properly, or for archival and backup purposes. The new law states that you may transfer (give, sell, lease, etc.) a copy of the program to someone else only if you transfer all of your copies and rights to the program. That is, you cannot give a friend a copy of a copyrighted program and still keep a copy for yourself.

As a matter of information, Radio Shack copyrights the Microcomputer News to prevent unauthorized commercial use of the material in the newsletter. You as an individual are free to use the information for your private benefit and to pass the information (including programs) along to other individuals for their private use.

If you provide us with a program for publication, and we are contacted by a commercial organization that wishes to use the program (this has happened recently) we will refer the interested parties to you (if you give us an address and/or phone number). You are then free to make whatever arrangements are satisfactory between you and the persons interested in your program.



View From the 7th Floor

by Jon Shirley, Vice President Computer Division

In the past I have avoided commenting on the magazine 80 Microcomputing which has often seemed to have a love/hate relationship with Radio Shack. But this month's issue (March) had so much neat stuff that I could just not resist the temptation to write a little about this amazing issue. First off, in the publishers remarks, is a very nice and fair column, which in summary says that despite our warts, the TRS-80 is still the best micro to own. Thanks Wayne, we agree 100%. I might also mention that the same column commented on Tandy overseas and stated that we are number three in sales. Sorry Wayne, but we are definitely number two and if the reputed number one published their unit sales we might even be number one.

But what else is in this issue? Lots! For example there is an interview with Mr. Kornfeld and another with our new president, Mr. John Roach. Both well worth reading. And there is more. There is a great rebuttal by Mr. H. C. Pennington, author of "TRS-80 Disk and Other Mysteries," to a reader's letter and Harv, we do not hate you in Fort Worth. We may disagree with you, TRSDOS 2.3 has been running a lot of business software for a long time with no problems, but every one is entitled to his/her opinions.

A lot of the issue is devoted to disk operating systems. One "DOS Talk" compares TRSDOS and 5, that's right 5, other DOS's. Do any other computers have 6 available operating systems? Thanks for the nice comments in that article too, TRSDOS 2.3 does work exactly as the manual says, first time, every time and special thanks for outstanding rating of our documentation. It's even better for the Model III and Color Computer. And finally there is an article on the author of the original Model I TRSDOS. While that was a complicated story which is probably of little interest to anyone and one we would like to forget, I would like to point out that the author was paid by Shugart as called out in his contract.

Part of the reason for commenting on this issue relates to a letter from a very angry Model I owner who wanted to know why we did not produce those 5 DOS's or why we did not write Mr. Pennington's book etc., etc. I suppose the major reason is resources. We may look large but back then we did not have a large staff and we were very busy writing, in house with our staff, Model II TRSDOS. When we could not get the author to fix TRSDOS 2.1 we stopped that project to produce 2.2 and then 2.3 And there is a difference between the two and it relates to how the real time clock interrupts are handled during disk access. But the real point is that there were books done and there were other disk operating systems produced and that forms part of the support for the TRS-80. One important measure of the acceptance of any computer is how much outside support it has. It is unrealistic to assume that any computer company can produce all the software its users desire. Our big blue monster made by the world's biggest computer company (who spends more on R and D than the total sales of any other company) runs a sort routine done by an independent software company. And guess what, so do over 50% of the other owners of that MAIN FRAME. All in all the TRS-80 has more support, both from Radio Shack and the outside world, than any other micro.

I recently spoke in New York on a panel that was part of a conference called "Information Utilities '81." My theme was that communications is the key to the "home computer." The afternoon session featured speakers from CompuServe, The Source, and Dow Jones (plus another I did not have time to hear). There is really a lot happening with these databases and yes, this is another pitch for Videotex and CompuServe and Dow Jones. CompuServe has over 9,000 users as I write this and is growing all the time both in users and in information. Dow Jones has a huge database for the investor and I find I am using it daily. A lot more stores are now equipped to demo CompuServe and the Computer Centers should have Dow Jones up by now so take a look, it's free.

In March we delivered our first Videotex host computer system. This is a four drive Model II attached to a 16 line telephone multiplexer that we designed and are making here. With one of these, up to 16 Videotex terminals can access the data stored on the Model II disks . . . all at the same time and even when new data is being downloaded to the Model II. The first lucky customer is the Professional Farmers of America who publish an agribusiness (how is that for a buzz word) newsletter to over 75,000 subscribers. With this system they are equipping their customers with a Videotex terminal which can call in to the Model II from all over America and receive the newsletter electronically. The advantage, of course, is that the information is up to the minute and not dependent on our dear postal service.

In average use one TRS-80 Model II host can handle one to two thousand Videotex terminals. Want one of these for your business? The entire system (without the terminals) sells for a little under $20,000 and we are taking orders. Just write to Fort Worth if you are seriously interested. By the way this is not a two way system, the terminals can only receive data, they cannot send data to be stored in the Model II. Give us a little more time for that. And in closing I would like to remind you of the TRS-80 Source Book. We are going to print now with issue 3 with over 2,000 applications software programs for all the TRS-80's included. If you have a program you want to list, just go to any Radio Shack, buy a source book (just $1.95) and fill out the enclosed blank. It's still $10 per listing and your program will be listed for a year. Judging from the number of new listings from writers who were in the first issue it must be working for them.

Until next month.

Back in February of this year, we published a list of Radio Shack Computer Centers. It turns out that some of the addresses and phone numbers we gave you were wrong. Below are corrections for the incorrect information, along with the addresses and phone numbers of some of our newest Computer Centers:

New and Corrected Computer Center Numbers

* Corrections	# New Computer Centers		
* Arkansas,	Little Rock	Town & Country Shop C.	501/568-5694
# Delaware,	Wilmington	3847 Kirkwood Hwy.	302/999-0193
* Florida,	W. Palm Beach	2271A Palm Beach Lk. Blvd.	305/683-3100
#	Jacksonville	8252 Arlington Expway.	904/725-2594
# Illinois	Niles	8349 Golf Rd.	312/470-0670
* Louisiana	New Orleans	327 St. Charles Ave.	504/523-6408
#	Shreveport	1545 Line Ave.	318/221-5125
# Maryland	Catonsville	One Mile West S/C	301/788-3277
* Massachusetts	Worcester	7 Gold Star Blvd.	617/852-2030
# Michigan	Roseville	31873 Gratiot Ave.	313/296-6210
#	Flint	G3298 Miller Rd. Yorkshire	313/732-2530
# Minnesota	St. Paul	6th & Wabasha	612/291-7230
# Nevada	Reno	3328 Kietzie Lane	702/826-6327
* New Jersey	E. Brunswick	595 A Route 18	201/238-7142
# New York	Buffalo	839 Niagara Falls Blvd.	716/837-2590
# North Carolina	Charlotte	3702 Independence Bl.	704/535-6320
* Ohio	Cleveland	27561 Euclid Ave.	216/261-6900
# Pennsylvania	Scranton	206 Meadow Ave.	717/348-1801
# South Carolina	N. Charleston	5900 Rivers Ave	803/747-5580
# Tennessee	Knoxville	Cedar Bluff S/C	615/690-0520
# Texas	Houston	211-C FM 1960	713/444-7006
# Washington	Spokane	7706 N. Division #1	509/484-7000
* West Virginia	Huntington	2701 ½ Fifth Ave.	304/523-3527

Printer Codes — The Inside Story

The following major article on Radio Shack printer drivers and printers is running this month instead of the usual Product Line Manager pages for Model I/III, Model II, and Peripherals. These pages will return as usual in June.

Last month in the Peripherals article we gave you some introductory information about Radio Shack line printers and how the Models I and III handle information which is sent to printers. We also discussed the LPC/CMD program which is required by certain programs when they are used with newer Radio Shack printers. This month we would like to show you what happens when you "tell" the computer to print something on a lineprinter, and be more specific about the exact responses you can expect from the various Radio Shack line printers.

The reasons we want to do this is to help clear-up some misunderstandings, and to give serious programmers information which may help them write programs that perform as the programmer expects them to.

Our discussion will cover several main points:

When you LPRINT a character, . . .

1) What happens in the printer driver
2) What happens between BASIC and the driver
3) What happens at the printer

Anatomy of a Printer Driver

What is a Printer Driver? That little beast is a machine language routine used by the computer to send individual characters to a printer. This driver is used whether you want to send a character, a word, or a book to the line printer. The computer sends all information one character at a time. There is a certain amount of communication (via the status bits) between the computer and the printer so that the computer will not send characters faster than the printer can deal with them.

We will give you general information for each of the computers Model I, II, and III. Please note that the sequence may not match the exact steps that the computer follows, however the ideas are what is important.

We will use decimal code values, in particular:

10 (0AH) represents a Line Feed (LF)
11 (0BH) represents a Vertical Tab (VT)
12 (0CH) represents a Form Feed (TOF)
13 (0DH) represents a Carriage Return (CR — Which in a Standard Radio Shack printer is a new line (NL). Each time the printer executes a CR, it also executes a line feed LF).

Also, the following words are defined:

Increment — add one to the current contents of the particular memory location and put the new value into that memory location.

Set or Reset — Place the value indicated into the memory location indicated.

In addition, there are two addresses in the Model I and four (the two from Model I plus two more) in the Model III which we need to be aware of:

16424 — The maximum number of lines, plus one, to print on a page.
16425 — The number of lines already printed on the current page (plus one in the Model III).
16426 — (Model III only) The number of characters already printed on the current printer line.
16427 — (Model III only) The maximum number of characters to print on each printer line.

Model I/III Memory Locations

Location	Initial Values	
	Mod I	Mod III
16424	67	67
16425	0	1
16426	XX	0
16427	XX	255

XX — In the Model I these locations are used for different purposes than in the Model III, and we are not concerned with those purposes here.

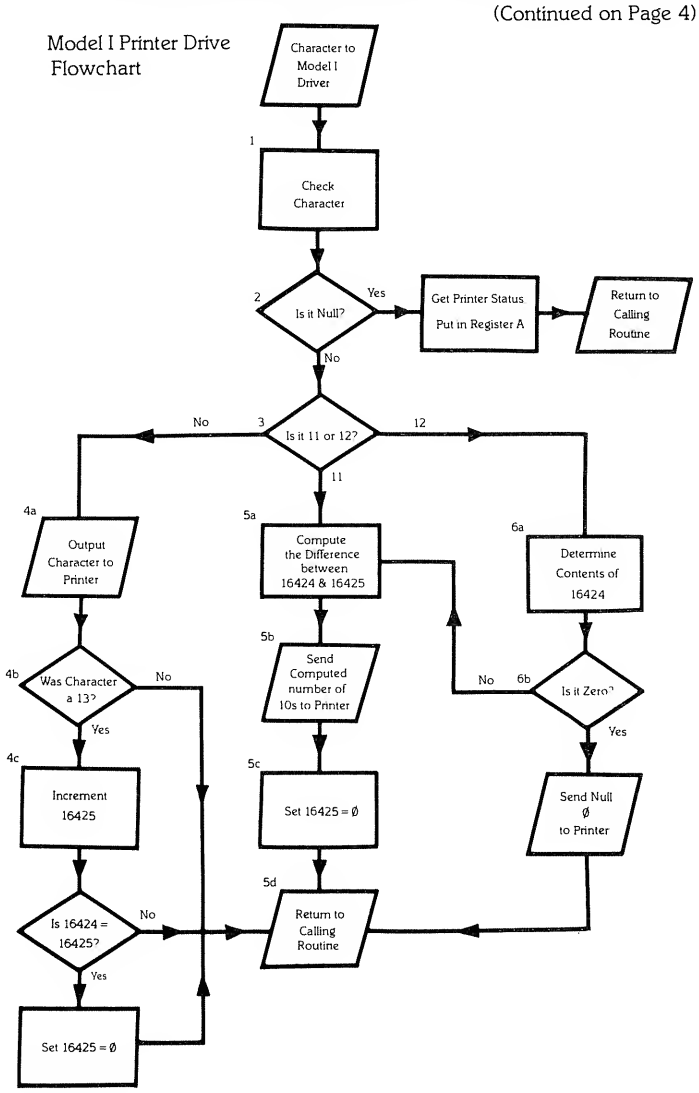
Model I

When the Model I system ROM printer driver receives a character from BASIC or from a machine language routine, to be sent to a lineprinter, the following major events happen (Flowchart 1 is keyed to these explanations):

1) The character is checked to see if it is a NULL (00 dec).
2) If the character is not a null, go to step 3, otherwise put printer status in the A register, and RETURN to calling routine. The null is NOT sent to the printer.
3) Check to see if the character is an 11 or 12. If it is an 11, go to step 5, if the character is a 12, go to step 6. If it is neither an 11 nor a 12, go to step 4.
4) a. Send the character to the line printer. Note that the ROM driver does not check to see if the character might

(Continued on Page 4)

Model I Printer Drive Flowchart



have been either a 10 or 13 before outputting it to the printer.

b. After the character is output to the printer, it is checked to see if it was a 13. If the character was not a 13, go to step 4d.

c. The character was a 13, increment memory location 16425. Memory locations 16424 and 16425 are then compared. If they are equal, 16425 is reset to zero. Whether 16425 is reset or not, go to step 4d.

d. Return to the calling routine.

5) The character is an 11, we want to perform a move to the top of the next page (TOF). Note: TOF is the only type of vertical tab or form feed which is supported in a Radio Shack lineprinter.

a. Compute the difference between the contents of memory location 16424 and the contents of memory location 16425.

b. Once this value has been computed, the computer sends that number of character 10s to the printer.

c. The memory location 16425 is set to zero (0).

d. The computer returns to the calling routine.

6) The character is a 12, we again want to move to the TOF. The method is a little different than shown in step 5.

a. The contents of memory location 16424 is determined.

b. If it is a zero, the computer sends a character 0 (null) to the printer. The computer returns to the calling routine.

c. If it is not a zero, then the procedure outlined in step 5 is used to perform the TOF.

Model III

In the ROM printer driver, the following major events occur: (Flowchart 2 is keyed to these explanations)

1) Is the character a control code (<32), an alphanumeric character (31 < character < 128), or a special character (>127)? If the character has a value greater than or equal to 32 then check to see if the value is greater than 127. If the value was greater than 127, go to step 7. If the value was greater than or equal to 32 but less than or equal to 127, go to step 6, otherwise go to step 2.

2) If the character is a 13 then go to step 5.

3) If the character is NOT a 12 then go to step 7.

4) The character is a 12, we want to perform a move to the top of the next page (TOF).

a. Compute the difference between the contents of memory location 16424 and the contents of memory location 16425.

b. Once this value has been computed, the computer sends that number of character 10s to the printer.

c. Reset memory location 16426 to zero.

d. Memory location 16425 is set to one (1).

e. The computer returns to the calling routine.

5) The character is a 13.

a. Get the value in 16426.

b. Is that value a zero? If it is a zero, go to step 7.

c. The value was not a zero, change the 13 to 10 and go to step 7.

6) a. Subtract 32 from the value of the character.

b. Store new character from look-up table.

c. Get the value in 16427.

d. If it was 255 then go to step 7.

e. If the values in 16426 and 16427 are not equal, go to step 7.

f. Output a character 13 to the printer, reset 16426 to zero.

g. Go to step 7.

7) a. Output the character to the printer.

b. Increment 16426.

c. If the character was a 10 or a 13 then go to step 8.

d. Return to calling routine.

8) a. Reset 16426 to zero.

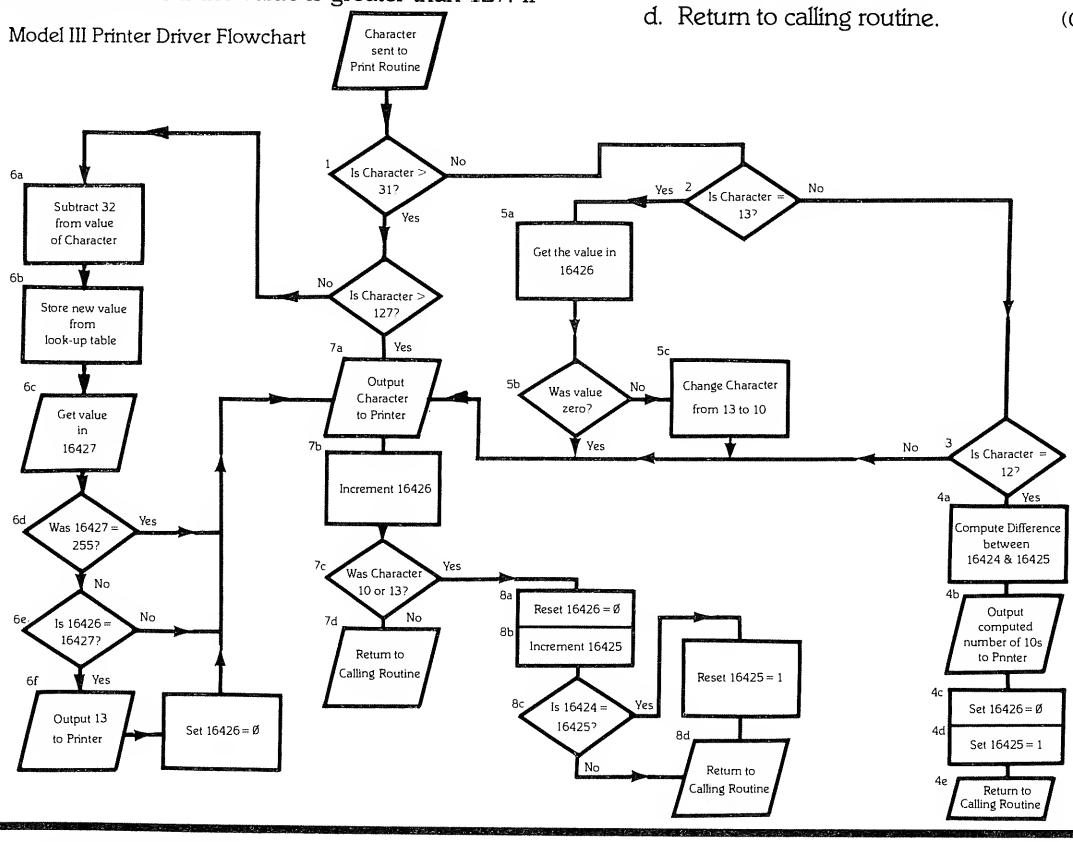
b. Increment 16425.

c. If 16424 and 16425 are equal (TOF) then reset 16425 to 1.

d. Return to calling routine.

(Continued on Page 5)

Model III Printer Driver Flowchart



Model II

The basic information about what the System printer driver does is contained in the Model II manual. Of particular interest is the Technical information section on lineprinter SVCs (TRSDOS Chapter 4) and the section on the FORMS command (TRSDOS Chapter 2).

Here is a summary of that information:

We will use the following abbreviations:

PL — Length of a page in lines
LP — Maximum number of lines to be printed on one page.
CL — The maximum number of characters to be printed on a single line.

Model II allows several options while using lineprinters:

1. Parallel or serial printers may be used.
2. Using an SVC (95) you can set current line and current character counts to a desired value.
3. Use a "Transparent mode" which sends all characters directly to the printer without any translation. Note: BASIC can not send a code 10 to the printer driver.
4. Use a "Dummy" mode which tells the computer to ignore all printer output.
5. Select whether or not to have an automatic line feed after each carriage return. Note: this is for printers which do not have the Radio Shack CR/LF feature and require that an LF be sent after each CR to get a new line.

Most of these options can be selected using the TRSDOS FORMS command. They can also be selected by using the appropriate SVC call.

Actions taken by the printer driver:

1) If PL and LP are the same, and PL does not equal zero the automatic Top of Form (TOF) is disabled. In this situation, if a character 11 or 12 is output, the printer driver will issue the proper number of 10 or 13 characters to advance the paper to TOF.
2) If PL equals zero, then LP must also equal zero. When PL and LP are zero, the characters 11 and 12 are sent directly to the printer. This allows you to take advantage of TOF and Vertical Tab features which your printer may have.
3) If PL is greater than LP, and neither of them is zero, an automatic TOF will be executed when the number of lines printed on a particular page equals LP. TOF is done by sending carriage return/line feeds to the printer.
4) If CL is set to zero, the tab character (9 dec — HT) will be sent directly to the printer, and the CL counter will be incremented by one. In addition, no automatic carriage return is performed.
5) If CL does not equal zero, the 9 character (HT) is translated into one to eight spaces to perform the tab function, and an automatic carriage return will be performed when the number of characters sent to the printer equals CL.
6) The following decimal codes are "intercepted" by the printer driver and translated as indicated:
   - 9 — HT — From one to eight spaces are sent to the printer.
   - 11 — VT — TRSDOS sends enough carriage returns or line feeds to the printer to advance to the top of the next form.
   - 12 — FF — Same as VT (11).
   - 13 — CR — When the current line is empty (no characters printed since the last carriage return or line feed), the 13 is translated to a 10. In the auto line feed mode, a 10 is sent after each 13.
   - 141 — TRSDOS sends a carriage return to the printer. In the auto line feed mode, using this code allows you to send a carriage return WITHOUT a line feed. Remember that the normal Radio Shack BASIC CR includes an automatic LF.

What happens between BASIC and the printer driver?

In the Models I and III, BASIC outputs anything you give it (in the form of an LPRINT statement) directly to the printer driver without translation except CHR$(10) (a line feed) which BASIC translates to a CHR$(13) which is a carriage-return (NL in Radio Shack standard printers). In addition, BASIC adds a CHR$(13) to the end of any LPRINT statement which does not suppress the CHR$(13) with either a trailing semicolon (;) or a trailing comma (,).

In the Model II, BASIC does not send the character 10 to the printer driver at all. Also, the character 9 (HT or horizontal tab) is translated into one to eight spaces. Model II BASIC, like Model I and III, adds a CHR$(13) to the end of any LPRINT statement which does not have a trailing semicolon (;) or trailing comma (,).

Please note that you can not send a character ten (CHR$(10)) to the printer driver from BASIC in Models I, II, or III.

How does this affect your programs?

If you are programming on a Model II, you should not have any real problems. This is mainly due to the fact that the Model II TRSDOS FORMS command will allow you to send exactly what you want to your printer, except the CHR$(10) from BASIC.

If you are programming on a Model I or a Model III, the primary "problems" which you can expect to encounter as you interface with a lineprinter are:

1. Improper Top-of-Form.
2. Too few line feeds.
3. In certain conditions, the printer may "spew" 255 line feeds.

The following procedures have been developed by our applications programming staff to solve the first two problems (This is a review of information we gave you in April):

To ensure proper action during a Top of Form, follow this procedure:

A. Set the line counter to 1 at the beginning of the program by using the following instruction:

```
POKE 16425,1
```

B. To do a TOF, follow this procedure:

```
LPRINT CHR$(12);:POKE 16425, 1
```

(Please note that the semi-colon (;) following the CHR$(12) is extremely important)

To ensure that you get the proper number of line feeds:

A. Do NOT use CHR$(138). If you have used CHR$(138), change it to CHR$(32). CHR$(32) is the code for a blank or space. CHR$(138) was originally suggested as a way to "sneak" a CHR$(10) to our original lineprinter (LP I). It works well with that printer. However, if your software might be used with any other printer, do not use it. Also do not use LPRINT without something to be printed. If you need a single line feed, use LPRINT CHR$(32) or LPRINT" ".

B. If you are going to use LPRINT CHR$(13) to generate a single NL, use it with a trailing semi-colon (;). Remember that BASIC adds a CHR$(13) to the end of any LPRINT statement which does NOT have the trailing semicolon or comma.

Our application programmers assure us that these procedures will work with both Models I and III, all ROM versions and all "known" printers.

The third problem occurs in both the Model I and Model III if the following conditions are met:

1. Memory location 16424 and 16425 are equal.
2. A TOF (Model I — CHR$(11) or CHR$(12), Model III — CHR$(12)) is sent to the printer driver. (Continued on Page 6)

This situation does not happen very often, in fact you would probably have to be POKEing around with 16424 and 16425, but it is possible, and you should be aware of it. You can prevent this error from occurring by making sure that memory locations 16424 and 16425 are not equal before you send a CHR$(11) or CHR$(12) to the printer. Under normal circumstances they will never be equal.

What do the printers do?

Printers have come a long way in a short time, and we see every indication that improvements are going to continue. When we compare our newer printers like the Line Printers VI and VII to our first lineprinter, the friction-feed, roll paper only printer that had a mechanically adjustable print head and a noisy motor that ran all the time, we find a lot of improvement in only three short years.

Because of these improvements, Radio Shack is dividing its printers into three categories for purposes of character generation and code response: Data Processing, Word Processing, and Graphics. Each of these three categories of printer will have its own "standard" response to codes sent to it from the computer. Our purpose is to create classes of printers so software can be designed to work with a particular level and class of printer rather than with a specific printer. As the number of different printers which are available increases, and as the features of each printer continue to vary it can be very difficult for both the programmer and the end user to understand the type of printer which should be used for a particular piece of software.

Some printers may use only one of these standards, and other printers might use all three. Further, some printers will contain only subsets of these standards, and (especially in our older printers) may respond to codes in a manner which is different than indicated by the standard. For this reason, we are presenting a chart of all current Radio Shack printers and their control code responses. We have also included information on how our various printers fit with other portions of the new standards.

Please realize that this standard is NEW and is not fully implemented on any printer which is currently in the Radio Shack Printer line. This standard tells you what we expect printers to do in the future. We are also including information on how the printers we have had up to this time fit with this new standard.

Data Processing Printer

The data processing capabilities of a printer are best used when we need to print a large quantity of information at the best (or fastest) possible speed.

In a Data Processing printer, dot matrix characters are acceptable. Fixed pitch 5 characters per inch (CPI) elongated, 10 CPI ordinary, and condensed print (typically 16.7 CPI) will all be available on a full Data Processing printer. The condensed character font allows you to print 132 characters in an eight inch print line. Underlining, proportional spaced character fonts, other character sets and fonts are not required and may or may not be present in a Data Processing printer. A typical Data processing printer will not have backspace capabilities.

In a data processing printer, you will be able to mix ordinary (10 CPI) and elongated (5 CPI) characters on a single print line. You may not be able to mix ordinary and condensed characters on the same line.

All line pitch control commands are stored but not executed by the printer. Upon receipt of the next LF code (0A Hex, 10 Dec) the paper is advanced in accordance with the stored pitch values.

A full performance Data Processing printer will have the standard Special character set (Table 2) and block graphics (Table 3) in addition to the 96 standard ASCII characters (Table 1).

Word Processing Printer

A Word Processing printer is used when we are less concerned with speed and are more interested in having neat "business-like" characters which will be fully formed or high density dot matrix characters. One or more fixed pitch character fonts may be available. These character fonts may also include proportional spaced characters. A Word Processing printer will have backspace, subscript, and superscript capabilities.

In a Word Processing printer, with appropriate software, you will be able to mix all character types on the same print line if you need to.

All line pitch control codes are accepted and a line feed is executed (with the appropriate distance and direction) upon receipt in Word Processing modes. One of the purposes of this change (from the Data Processing mode of storing pitch information) is to make actions such as subscripting and superscripting relatively easy.

An extended Special Character set (Table 2) will be available in a full Word Processing printer along with the standard 96 ASCII characters (Table 1). The Block Graphic characters are not required in a Word Processing printer.

Graphics Printer

A Graphics (or Bit Image) printer is used when we want to produce high density pictures or graphs using a printer. The current dot matrix graphics printer (LP VII) allows you to control seven pins in the dot matrix print head (dot addressable) in order to achieve maximum resolution. The bit image mode can be

(Continued on Page 7)

Data Processing Control Code Standard—Explanations and Responses

HEX	DEC	SYMBOL	Explanation	I	II	IIA	III	IIIA	IV	V	VI	VII	WP5Q	QUME	DWII	QP	QPII	PLOT
0A	10	LF	Execute CR/LF	+	+	+	+	+	+	+	+	+	+	+	+	+		+
0D	13	CR	Execute CR/LF	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
1B 36	27 54	ESC 6	Select 6 LPI				+	+		+	+		+	#				
1B 38	27-56	ESC 8	Select 8 LPI				+	+		+	+			#				
1B 1C	27 28	ESC FS	Select 12 LPI						+	+	+			%	+			
1B 0E	27 14	ESC SO	Start Condensed			*			*	+	+					$		
1B 0F	27 15	ESC SI	End Condensed			*			*	+	+					$		
1F	31		Start Elongated		@	@	+	+	@	+	+	+				+	&	
1E	30	RS	End Elongated		@	@	+	+	@	+	+	+				+		
12	18	DC2	Select Graphics									+						
13	19	DC3	Not Expected															
14	20	DC4	Select Word Proc.															

* ESC DC4 to select condensed, ESC DC3 to select ordinary characters
@ ESC SO to select elongated, ESC SI to end elongated
$ Use GS (Dec 29) to select 20 CPI, RS (Dec 30) to select 10 CPI, US (Dec 31) to select 5 CPI
% ESC U is ½ Forward LF (12 LPI)
# See Qume manual for details
& Use SI (Dec 15) to get elongated characters.
QP II automatically returns to ordinary characters at end of line.

a very powerful tool for individuals who have graphics requirements.

Graphics Control Code Standard—Explanations and Responses

HEX	DEC	SYMBOL	Explanation	LP VII
0A	10	LF	Forward LF	+
0D	13	CR	Execute CR/LF	+
1E	30	RS	End Graphics	+
1B 0E	27 14	ESC SO	Start Elongation	+
1B 0F	27 15	ESC SI	End Elongation	+
1B 10	27 16	ESC POS	Positioning	+
1C	28	FS	Repeat Data	+

Brief note: some current (LP III) and future printers which are intended for sale in Japan as well as the United States and Europe have or will have the Japanese Kana character set. Where the Japanese Kana set is available, it may or may not require a repair center to select. The Kana set will replace the special character set as well as using some additional (otherwise unused) codes.

Control Codes

According to the American Standard Code for Information Interchange (ASCII), there are 34 single byte control codes and multiple byte ESC sequences in addition to a printer's printable character set. The new Radio Shack printer standards recognize 23 of these codes and sequences, and others may be added as printers get smarter. There are two possible actions which can be taken by the printer when it receives unprintable or non-active codes. The unprintable or non-active code can be ignored, or it can cause a special character to print depending on the type of printer and its current operating mode. Please note that the concept of a special character for unprintable codes is not yet implemented in any Radio Shack printer. Current printers ignore the character or print a blank, depending on the printer.

The following charts and tables will show the new Radio Shack printer standards. This is how we expect future Data Processing, Word Processing and Graphics printers to respond. Most of these charts will also detail how current Radio Shack printers respond to these new standards.

Standard Alphanumeric characters

The standard alphanumeric characters in Data Processing and Word Processing modes are the 96 standard ASCII (American Standard Code for Information Interchange) characters. These characters are shown in Table 1.

In the Graphics or Bit Image mode, all characters starting from 32 Dec (20 Hex) to and including 127 Dec (7F Hex) are ignored by the printer.

(Continued on Page 8)

Table 1

96 ASCII ALPHANUMERIC CHARACTER SET

HEX	DEC	CHAR	HEX	DEC	CHAR	HEX	DEC	CHAR	HEX	DEC	CHAR
20	32	SPACE	38	56	8	50	80	P	68	104	h
21	33	!	39	57	9	51	81	Q	69	105	i
22	34	"	3A	58	:	52	82	R	6A	106	j
23	35	#	3B	59	;	53	83	S	6B	107	k
24	36	$	3C	60	<	54	84	T	6C	108	l
25	37	%	3D	61	=	55	85	U	6D	109	rn
26	38	&	3E	62	>	56	86	V	6E	110	n
27	39	'	3F	63	?	57	87	W	6F	111	o
28	40	(	40	64	@	58	88	X	70	112	p
29	41	)	41	65	A	59	89	Y	71	113	q
2A	42	*	42	66	B	5A	90	Z	72	114	r
2B	43	+	43	67	C	5B	91	[	73	115	s
2C	44	,	44	68	D	5C	92	\	74	116	t
2D	45	–	45	69	E	5D	93	]	75	117	u
2E	46	.	46	70	F	5E	94	^	76	118	v
2F	47	/	47	71	G	5F	95	_	77	119	w
30	48	0	48	72	H	60	96	`	78	120	x
31	49	1	49	73	I	61	97	a	79	121	y
32	50	2	4A	74	J	62	98	b	7A	122	z
33	51	3	4B	75	K	63	99	c	7B	123	{
34	52	4	4C	76	L	64	100	d	7C	124	|
35	53	5	4D	77	M	65	101	e	7D	125	}
36	54	6	4E	78	N	66	102	f	7E	126	~
37	55	7	4F	79	O	67	103	g	7F	127	Ignored

Word Processing Control Code Standard—Explanations and Responses

HEX	DEC	SYMBOL	Explanation	LP I	II	IIA	III	IIIA	IV	V	VI	VII	WP50	QUME	DWII	QP	QPII	PLOT
0A	10	LF	1/6" For. LF	+	+	+	+	+	+	+	+	+	+	+	+	+		+
1B 0A	27 10	ESC LF	1/6" Rev. LF						+					+	+			
1B 38	27 56	ESC 8	1/8" For. LF				SEL	SEL		SEL	SEL							
1B 1C	27 28	ESC FS	1/12" For. LF						+	SEL	SEL			%	+			
1B 1E	27 30	ESC RS	1/12" Rev. LF						+					%	+			
0D	13	CR	Execute CR/LF	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
08n	08n	BS n	Backspace						+				+	+	+			+
1B n	27 n	ESC n	Incrernental space						+					#	+			
0F	15	SI	Start Underline						+	+					+	+		
0E	14	SO	End Underline						+	+					+	+		
1F	31		Start Elongated		@	@	+	+	@	+	+	+				$	&	
1E	30	RS	End Elongated		@	@	+	+	@	+	+	+				$		
1B 11	27 17	ESC DC1	Sel Prop. Char.						+						+			
1B 13	27 19	ESC DC3	Sel Ordinary		+	+			+									
1B 14	27 20	ESC DC4	Sel Condensed		+	+			+									
12	18	DC2	Select Graphics									+						
13	19	DC3	Select DP Mode															
14	20	DC4	Not Expected															

SEL—Line feed response is selected but not executed until printer receives 0A Hex LF.
% ESC U for 1/12" Forward LF, ESC D for 1/12" Reverse LF.
# See Qume manual for details
@ ESC SO to select elongated, ESC SI to end elongated
$ Use GS (Dec 29) to select 20 CPI, RS (Dec 30) to select 10 CPI, US (Dec 31) to select 5 CPI
& Use SI (Dec 15) to get elongated characters.
QP II automatically returns to ordinary characters at end of line.

1B 15 27 21	RESERVED
1B 16 27 22	RESERVED
1B 17 27 23	RESERVED
1B 18 27 24	RESERVED
1B 19 27 25	RESERVED
1B 1A 27 26	RESERVED

Unprintable Codes

In the new standards, in both Data Processing and Word Processing modes, the following ranges are generally considered to be "Unprintable" and will result in the printer printing a special "unprintable code" symbol — X.

Unprintable Code Ranges:

128 Dec (80 Hex) through 159 Dec (9F Hex)
192 Dec (C0 Hex) through 223 Dec (DF Hex)

In addition, if you are using Proportional spaced characters, the following code range is also unprintable:

224 Dec (E0 Hex) through 254 Dec (FE Hex)

In the Graphics mode, there are no "unprintable" codes, just a lot of "ignored" codes. Except for the seven control codes listed in the Control code chart, ALL codes from 00 Dec (00 Hex) through 127 Dec (7F Hex) are ignored in the Graphics mode. Since this includes all "seven bit" codes, eight bits (serial or parallel) are required to use a printer's graphics (Bit Image) mode.

Special Character Set

Table 2 shows the 25 characters which form the new Special character set. It also shows each printer which has any of the characters and the decimal value to use for that character. This character set will be available on all future Data Processing and Word Processing printers. In addition to the standard special characters, Word Processing printers will have five additional characters to complement the standard set. These five additional characters are (and the printers they are available on):

Character	QUME	DW II
´	—	167
®	92	170
¾	—	173
™	126	169
¨	—	190

Table 2

Standard Special Character Set

Character	LP V	VI	QUME	DW II	Character	LP V	VI	QUME	DW II
à	161	161	—	128	Ä	177	177	—	219
ç	162	162	—	156	Ö	178	178	—	220
£	163	163	—	163	Ü	179	179	—	221
µ	165	165	—	165	¢	180	180	—	222
°	166	166	96	166	ä	182	182	—	251
▼	167	167	—	—	ö	183	183	—	252
†	168	168	125	168	ü	184	184	—	253
§	169	169	123	192	β	185	185	—	254
©	171	171	94	171	é	187	187	—	187
¼	172	172	—	172	ù	188	188	—	188
½	174	174	—	174	è	189	189	—	189
¶	175	175	124	175	f	191	191	—	191
¥	176	176	—	204					

Block Graphics Characters

The Block Graphic characters are 31 special "graphic" symbols which are included as part of a Data Processing printer's standard character set, and are not part of the Graphics Mode. These block graphic characters are optional in Word Processing printers, and are not available in the Graphics or Bit Image mode. Where the block graphic symbols are available, you can expect to see them only with fixed pitch character fonts and the various manipulations of those fonts.

These symbols are currently available only in the Line Printers V (26-1165) and VI (26-1166). The symbols and their codes are shown in Table 3.

DETAILS

We have now described, in general what we expect our new printers to be able to do as they become available, as well as given you an indication of how our current printers fit the new standards. Now we are going to expand on some of the ideas we have given you, fill in some of the dark areas, and generally make all of this more useful to you (we hope).

Table 3

Block Graphic Symbols

Hex	Dec	Symbol	Hex	Dec	Symbol	Hex	Dec	Symbol	Hex	Dec	Symbol
E0	224	(Blank)	E8	232	▀	F0	240	⌐	F8	248	⊥
E1	225	▘	E9	233	▌	F1	241	¬	F9	249	┤
E2	226	▝	EA	234	▐	F2	242	┐	FA	250	┼
E3	227	▀	EB	235	▛	F3	243	┬	FB	251	◤
E4	228	▖	EC	236	▜	F4	244	├	FC	252	◢
E5	229	▚	ED	237	▙	F5	245	│	FD	253	◥
E6	230	▞	EE	238	▟	F6	246	└	FE	254	◣
E7	231	▄	EF	239	▄	F7	247	┘			

Backspace (BS n)

Backspace is required in Word Processing printers but not in Data Processing printers. Backspace is not available in the Bit Image mode.

Where backspace is available, it will usually take one of two forms. The backspace command may be a single byte command (BS) which simply tells the printer "backspace." This is the type of BS present on our daisy wheel printers. With this type of BS command, either the printer only supports one size character (which means that the backspace distance is fixed), or the printer has some internal mechanism for "remembering" what characters have been printed, and how far the print head needs to be moved to backspace over them.

The second type of backspace uses the "BS n" form of the command. For instance, dot matrix printers which have backspace capabilities can be backspaced from one to 255 incremental or micro spaces (dots) by using the sequence BS n (08 n hex — 08 n dec, n is the hex or dec equivalent of a value from 1 to 255 as appropriate and as limited by the particular printer being used—LP IV only allows 127 micro-spaces). When the BS n code is received, the printer will subtract n dot columns from the current print head position and move the print head accordingly.

If n equals zero, no backspacing is done. If the value of n is greater than the number of dot columns the print head has moved from the beginning of the line (Home position), subsequent printing will begin at the beginning of the line.

The BS (08) code is ignored in the Bit Image mode and the n will be treated as an independent graphics data byte. This should be considered an error condition and you should program to prevent this from occurring.

If you wish to backspace one character or space, the value of n should equal the appropriate value for the character set and printer being used. On a printer using monospaced characters, n will be the number of incremental spaces defined by your printer manual for that particular character size. If you are using proportional spaced characters, the value of n will vary from character to character, requiring that your program "know" exactly what characters must be backspaced over. This may require that your program contain some type of "look-up" table containing the horizontal micro-space counts for each character in the printer's character set. Whether you use a look-up table or not, if you are going to use backspacing which has a BS n format, you must have some method of telling the printer how many micro-spaces to back-up.

Line Feeds (LF, ESC LF, ESC FS, ESC RS, ESC 6, ESC 8)

The only true line feed command is the LF (0A Hex — 10 Dec). The other commands which are referred to as "line feed" commands are really pitch control commands.

In a Word Processing printer, any of the "line feed" commands will result in an appropriate movement of the carriage, but we would like you to think of two separate things happening. First, the printer is given the pitch control information and second a line feed occurs which executes the pitch control information which was provided (after printing the current contents of the print buffer if there are characters in the buffer). (Continued on Page 9)

In the Data processing mode, this action is easy to see since the only "active" line feed command is LF (0A Hex – 10 Dec). All other line feed codes, such as eight pitch (ESC 8), are stored by the printer until an LF is received. When the LF command is received, the entire contents of the print buffer are printed, then the specified line feed action is taken using the stored pitch information. If no other line pitch information has been received, the LF command will cause a 6 Lines per inch (6 LPI) forward line feed. See Table 4.

There are several things which you should realize about line feeds. First, you can not get a LF (Dec 10) from Model I, II, or III BASIC to the printer. The LF is generated by the printer when the computer sends a CR. (This is not absolutely true, there are times when the COMPUTER will send LFs to the printer, the computer just won't let you do it. Also, if you disable the automatic CR/LF function, in those printers which allow this to be disabled, the LF after CR will disappear and you will have to have some method of sending LFs to the printer.)

The second thing to realize is that ESC 6 and ESC 8, require you to send the ASCII value for the characters 6 and 8 (decimal 54 and 56 respectively) and not the decimal values 6 and 8. This can be a little confusing if you look at the incremental space commands which also appear to have an ESC 6 and an ESC 8. The difference is that for incremental spacing, the ASCII symbols are ESC ACK and ESC BS and you send the printer decimal 27 6 and 27 8 respectively. Don't get these two sets of commands confused. We have tried to help by listing the decimal values in all the charts and tables in this article.

The third thing to be aware of is that in the Data Processing mode, the pitch control commands will remain set until you change them. This can cause problems if you want to do a single reverse line feed, but forget to reset forward line feeds after you get the one reverse line feed that you wanted. (Use ESC 6 to reset forward 6 LPI.)

In the Bit Image mode, the only line feed command available is LF (10 dec). This will advance the carriage some part of a normal line feed in the forward direction. For the LP VII this is 1/9th of an inch. In the Bit Image mode, all pitch control commands are ignored.

Table 4 is a summary of the line feed/pitch control commands for the Data processing and Word processing modes. Remember that the Bit Image mode only has one line feed (0A Hex – 10 Dec, which moves the carriage in the forward direction).

Table 4

LINE FEED DESCRIPTIONS

Hex Code	Dec Code	Symbol	Data Processing mode	Word Processing mode
0A	10	LF	This is the only active line feed in the Data Processing mode. The pitch control codes set the information (distance and direction), but action is not actually taken until an LF is received. When the LF is received, the text in the print buffer is printed, and the line feed is performed in accordance with the stored pitch.	Any text in the print buffer is printed and the paper is advanced (⅙ inch) forward. If the printer is set to do an automatic CR when a LF is received, the print head will move to the beginning of the next line, otherwise, the paper will simply move ⅙ inch forward and printing will continue on the next line without the print head returning and staying at the beginning of the next line.
1B 0A	27 10	ESC LF	The printer is set to do full (6 LPI) reverse line feeds. The reverse line feed is not executed. As soon as the ESC LF is set, the printer continues to receive data until a LF or CR is received, or a buffer full condition occurs.	Any text in the print buffer is printed and the paper moves (retracts) one full (⅙ inch) line in reverse.
1B 1C	27 28	ESC FS	The printer is set to do half (12 LPI) forward line feeds. The ½ forward line feed is not executed. The print buffer is not printed.	Any text in the print buffer is printed and the paper moves one half (1/12 inch) line feed forward.
1B 1E	27 30	ESC RS	The printer is set to do half (12 LPI) reverse line feeds. The ½ reverse line feed is not executed. The print buffer is not printed.	Any text in the print buffer is printed and the paper moves one half (1/12 inch) line feed in the reverse direction.
1B 36	27 54	ESC 6	The printer is set to do full (6 LPI) forward line feeds. The full forward line feed is not executed. The print buffer is not printed.	This sequence is ignored.
1B 38	27 56	ESC 8	The printer is set to do ¾ (8 LPI) forward line feeds. The ¾ forward line feed is not executed. The print buffer is not printed.	Any text in the print buffer is printed and the paper moves ¾ (⅛ inch) line feed forward.

Carriage Return (CR)

A CR code (0D hex – 13 dec) tells the printer to do two things. The first is to print the current contents of the printer buffer, the second is to execute a carriage return (return the print head to the beginning of the line). In most TRS-80 line printers the CR command also performs a third function, which is to advance the carriage to the next line (NL). See Table 5 for possible CR options.

Printers using this standard will have several options for the CR command. These options will be changeable either through a user accessible pair of switches, or they may have to be changed by your computer repair facility. Note: many early Radio Shack Printers are not changeable.

Table 5

CARRIAGE RETURN OPTIONS

Status	Description
When: CR = NL & CR ≠ CCR	When a CR code (0D hex – 13 dec) is received, printable character information is printed before execution of carriage return operation, then a line feed (One line (⅙ inch) forward in Word Processing mode, previously set line feed from printer memory in Data Processing mode, and a fixed portion of a forward line feed in bit image mode) is executed. Subsequent printable characters are printed from the start of the new line.
CR = NL & CR = CCR	Same as (CR = NL, CR ≠ CCR) in operation. But, if several CR codes are received successively without any printable characters, only the first CR is active. All succeeding CR codes with no intervening printable characters are ignored.
CR = CR & CR ≠ CCR	When a CR code is received, any printable characters in the printer buffer are printed before execution of the carriage return (no line feed is executed). Subsequent printable characters will be printed beginning from the left margin on the same line unless a LF command is received.
CR = CR & CR = CCR	When a CR is received, any printable characters in the printer buffer are printed and then a carriage return is executed (no line feed is executed). Subsequent printable characters will be printed beginning from the left margin of the same line unless a LF command is received. If more than one CR command is received with no intervening printable characters, only the first CR in the series will be executed. All subsequent CR commands received with no intervening printable characters will be ignored.

(Continued on Page 10)

Printer Code Response Chart

	Option 1		Option 2		Change-able ?
	CHR$(13) CR = NL	CHR$(13) CR = CR	CR = CCR *	CR ≠ CCR	
LP I	X		X		No
LP II	X		X		No
Early LP III	X			X	No
Later LP IIIA	X		X		Yes
LP IV	X		X		No
LP V	X			X	Yes
LP VI	X			X	Yes
LP VII	X			X	No
WP 50	X			X	No
DW II	X			X	No

If the printer response is changeable, the chart shows the way the printer is shipped from the warehouse.

These changes are documented in our Service Manuals. You should let the R/S service center adjust the machine as your warranties are voided when the machine is opened by unauthorized personnel. The ability to make these changes will be a feature of all future Radio Shack Dot Matrix printers as long as there is a perceived need.

*NOTE: This setting is provided to allow new printers to respond in a manner which is compatible with the response of the LP I. We do not recommend this setting for new printers.

Getting down to business:

The following conditions will cause the printer buffer to be printed. Also shown is what happens after each of these conditions is met.

No.	Conditions which cause Printing	Operation after the buffer is printed
1.	When the number of dot columns or characters (depending on how your printer counts) in the print buffer equals or exceeds the maximum number of dots or characters, for your printer and the character font you are using, the buffer will be printed. If the last character received caused the count to exceed the maximum, the buffer will be printed without that last character.	After the buffer has been printed, the overflow character (if there was one) will be placed at the beginning of the print buffer. Any new characters which are received will be added to the buffer and the process will continue. When the buffer is printed by the printer in this manner, wrap-around has occurred and the printer issued a NL command. The next line printed will begin at the beginning of the next line after the one just printed. You should remember that in the Data Proc. mode line feeds are performed using stored pitch information. This means that if wrap-around occurs in the Data Processing mode, the LF issued by the printer may be a partial LF and it could be in the reverse direction. Wrap-around IS an error condition!
2.	If the print buffer becomes full, but the total number of dot columns to be printed does not equal or exceed the maximum number of dot columns for the selected character style, the printer will print the characters in the buffer. The buffer will now be empty.	Characters which are received after the buffer has emptied in this manner will be printed on the same line as the characters just printed. (No carriage return or line feed is issued in this case and printing continues on the same line.)
3.	The buffer will be printed if there is at least one character in the buffer and a CR code is received.	If CR = NL (normal setting) has been selected, the subsequent characters printed will be printed at the start of the next line. If the CR = CR option has been selected, the characters subsequent to the CR will be printed beginning at the beginning of the same line (unless a LF is received). This will cause overprinting.
4.	The buffer will be printed if there is at least one character in the buffer and a LF code is received.	If LF = NL, subsequent characters printed will be printed at the start of the next line. If LF = LF then subsequent characters will be printed beginning in the next character position on the next line.
5.	The buffer will be printed if there is at least one character in the buffer and the next character is not received within approximately one second (the amount of time will depend on the printer).	After printing the buffer for this reason, subsequent characters will be printed on the same line. (No CR or LF will be executed.)
6.	The print buffer will be printed if there is at least one character in the buffer and a POS sequence is received.	After the printing is performed, the next character after the POS sequence will be printed on the same line, beginning from the position specified by the POS sequence (Overprinting will occur if the new head position is to the left of the old head position).
7.	In the Data Processing or Word Processing modes, if there is at least one character in the buffer and the BS command is received, the buffer will be printed.	The next characters printed will be printed on the same line, beginning at the starting point indicated by the BS command. Overprinting will occur.
8.	In the Word Processing and Data Processing modes, when at least one character is in the print buffer, and a character font selection command (ESC DC1, ESC DC3, ESC DC4) is received, the buffer will be printed IF the selection code received is for a font which is different than the current character font. If the selection code is for the current character font, no action is taken.	After the buffer is printed, characters received will be printed using the new character set. Printing will continue on the same line until a LF or CR is received. Note: Some Data Processing printers cannot handle this function and may issue either an automatic NL or CR. In general, you can change character fonts or manipulate a font (condense or elongate it) in Word Processing printers, but you can only Elongate a font in a Data Processing printer.
9.	In the Word Processing and Data Processing modes, if there is at least one character in the print buffer, and the Bit Image selection code (DC2) is received, the buffer will be printed.	After the buffer has been printed, the Bit Image mode is selected and subsequent characters are treated as Bit Image data. Printing continues on the same line until a LF or CR command is received.
10.	In the Bit Image mode, if there is at least one data byte in the print buffer, and the End Bit Image command (RS) is received, the buffer will be printed.	All Bit Image data in the buffer is printed, and the printer is set to Data Processing mode in 10 or 5 CPI depending upon whether the elongated characters are being used. If underlining was on before Bit Image mode was selected, it will still be on when Data Processing mode is entered. The next characters received will be printed on the same line (No automatic CR or LF is executed).

Note 1: Conditions 1 to 6 are conditions which may occur in any of the Data Processing, Word Processing or Bit Image modes.

Conditions 7 to 9 are possible in either Data Processing or Word Processing modes, but not Bit Image.

Condition 10 can occur only in the Bit Image mode.

Note 2: In the descriptions, "next line or NL" means a new line after a line feed operation (CR/LF).

In Data Processing mode, if the reverse line pitch is available and has been set in the printer memory, line feed operations will cause the paper to move in reverse.

Note 3: Repeat data as well as extra incremental spaces can cause a buffer full condition and force automatic printing of the print buffer. If this occurs, any of the repeat characters not printed on the first line will be printed at the beginning of the new line.

Wrap-around

Wrap-around is considered to be an error condition. That is, under normal circumstances wrap-around should never occur. If wrap-around does occur, and you did not intend for it to occur, you should check your program to find out why too much data was sent to the printer before a print command was issued.

As mentioned in the section on what causes the print buffer to be printed, the printer will automatically print the buffer contents when the buffer is full. The buffer full condition exists when

(Continued on Page 11)

the number of dot columns (or characters) to be printed equals, or exceeds the maximum number of dots (or characters) for the printer buffer in the selected character set.

The purpose of this automatic wrap-around feature is to prevent the loss of print data. You should be aware of three things with regard to automatic wrap-around.

First, if your program counts the number of lines which have been printed on a page (or uses the computer's automatic lines per page counter) this counter will not take the extra lines (printed because of automatic wrap-around) into account. It is the printer, and not the computer which issues the CR/LF for automatic wrap-around, and the computer has no way of knowing that automatic wrap-around has occurred unless your program keeps close track of the number of characters (or dot columns for proportional and big image data) that have been sent to the printer since the last CR operation.

Second, if you are printing more than one style character on a print line, computing the number of dot columns or characters which are represented in the buffer can be complicated. If you will be using more than one style character in a single line, use caution or you may get un-wanted automatic wrap-around.

The third thing you should be aware of relates to the Data Processing mode. When automatic wrap-around occurs in the Data Processing mode, the line feed operation is done in accordance with the stored line feed and pitch information. If you have set half reverse line feed in the Data Processing mode, and automatic wrap-around occurs, the line feed will be a one-half reverse line feed.

Underline (SI, SO)

In the Data and Word Processing modes, reception of the SI (0F hex – 15 dec) code by the printer will turn on the automatic underline feature and all subsequent characters will be underlined. Please note that some printers will underline spaces as well as characters, while others (such as the DW II) do not underline spaces, even if the underline feature is selected. The underline function will continue (except while in Bit Image mode) until the SO (0E hex – 14 dec) code is received by the printer.

Once the underline mode is set, it will remain set even though CR or LF commands are received. Further, while the underline mode will not work in the Bit Image mode, if underline was on before entering the Bit Image mode it will still be on when you exit the Bit Image mode.

Incremental Spacing

In the Word Processing mode, any character data can be left and/or right justified by using incremental spacing (or micro-spacing) commands. Incremental spaces are not required in Data Processing printers, and are ignored in the Bit Image mode.

By using the incremental spaces, text can be adjusted within a line to move the rightmost character to a specified margin. Incremental spaces can be inserted between words or characters within a word by using appropriate software. The current Incremental space commands are:

ASCII Symbols	Decimal	Hexadecimal	Function
ESC SOH	27 01	1B 01	One micro-space
ESC STX	27 02	1B 02	Two micro-spaces
ESC ETX	27 03	1B 03	Three micro-spaces
ESC EOT	27 04	1B 04	Four micro-spaces
ESC ENQ	27 05	1B 05	Five micro-spaces
ESC ACK	27 06	1B 06	Six micro-spaces
ESC BEL	27 07	1B 07	Seven micro-spaces
ESC BS	27 08	1B 08	Eight micro-spaces
ESC HT	27 09	1B 09	Nine micro-spaces

Positioning (ESC POS N1 N2)

Positioning is required in a Graphics printer, but not required in a Word or Data Processing printer.

When positioning is available, you can instruct the printer to move the print head to an absolute position relative to the "Home" or leftmost possible print position.

After receipt of the positioning command, subsequent printing will begin from the specified head position.

ESC POS(1B 10 hex—27 16 dec) tells the printer that it is to position the print head. N1 contains the two most significant bits of the new head position and N2 contains the 8 least significant bits of that new position. At this time N1 will always be a value from 0 to 3 (only the least significant two bits in N1 are used, the higher six bits are always zero). See the chart below (and your printer manual):

	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1
First Byte (N1)	0	0	0	0	0	0	A_{10}	A_9
Second Byte (N2)	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1

Note 1: Depending upon the printer you may not be able to move the print head to each possible dot column using positioning.

Repeat Data (FS N DATA)

The repeat data function is not required in Data or Word Processing printers, and is required in a Graphics printer.

If repeat data is available, a single character can be duplicated up to 256 times (depending on your printer) by using the FS N DATA command sequence. (If you are using the serial interface, this assumes you are using 8 bit interfacing. If you are using 7 bit interfacing (such as from the Color Computer) you can repeat a single character up to 127 times. FS is 1C hex or 28 dec. The N tells the printer how many times to repeat the character and will be a value from 00 - FF hex or 0 - 255 decimal. Zero (0) hex or decimal will cause 256 repetitions in all modes (including both 7 and 8 bit serial).

The DATA byte is the byte which will be repeated. This should normally be a character in the printer's character or graphics set. In Data or Word Processing, if the DATA byte is in the range from 0 to 1F hex or 7F to 9F hex, or FF hex, the character will be changed to a "X" unprintable code mark. In the Bit Image mode, if the most significant bit of the value is a zero the repeat command sequence will be ignored.

Graphics Mode

The graphics mode is the bit image or dot addressable mode. In the current bit image mode, you have control of seven pins in the dot matrix printhead. The pins to be used for printing a particular column are selected by the pattern of the least significant seven bits of the value you send to the printer (which is why we call it bit image). If you want to print only the top dot of the seven available dots, the seven bits would be 0000001. The most significant bit for graphics is always a one, so the value sent to the printer would be 10000001 binary, or 129 decimal.

The Graphic characters begin at Decimal 128 (80 Hex, 10000000 Binary) with none of the pins used and continue through 255 Dec (FF Hex, 11111111 Binary) with all of the pins used. The important thing to remember, when you are trying to visualize what a dot pattern looks like, is that the first digit (one-dot on or zero- dot off) starting from the right represents the top dot of the seven dot pattern. The second digit from the right is the second dot from the top, etc. until you have the seventh dot counting from the right which is the bottom dot in the column you will print. (See Drawing 1 for an example.)

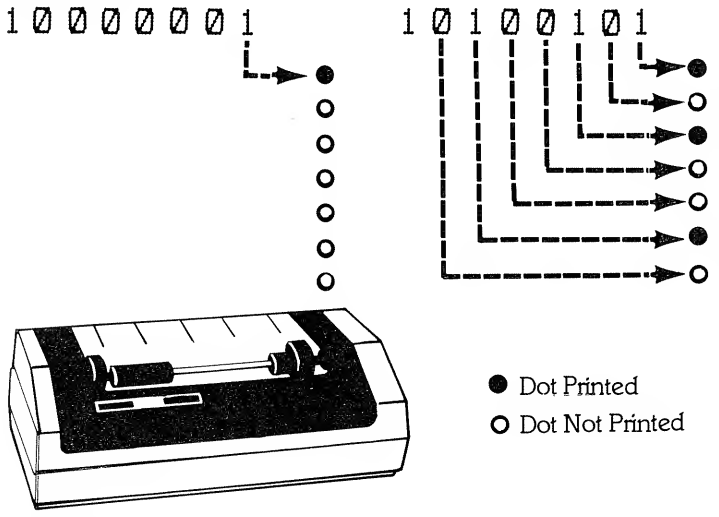
In general, each line of Bit Image graphics will take less vertical space (9 LPI vs 6 LPI in LP VII) than one standard line (remember, we are only printing seven dots). For this reason, Bit Image has its own line feed (Still Decimal 10, but less than one full line feed). The purpose of the special line feed is to let you create dense graphics without the vertical space between lines which you have in normal printing.

The Line printer VII (26-1167) is the only Radio Shack printer which currently supports the Graphics mode. In the Line printer

(Continued on Page 12)

VII graphics mode there are 60 horizontal dots per inch and 63 vertical dots per inch. This gives you 3780 dots per square inch of resolution. Watch out Picasso!

Drawing 1



Ignored and Unprintable Codes

In each and every printer there are valid codes which the printer may receive, but for which the printer may not have a defined control function or printable character. For example, ASCII defines Decimal code 7 as being a bell. Radio Shack has sold only one printer, the Quick Printer, which had a bell function. If you should happen to be running a different printer with some software originally written for a Quick Printer, the printer might receive the Bell code 7. What is the printer going to do with this "unknown" (to it) code?

In past printers there has not been any real standard for action to be taken by a printer when it gets a code which is not in the printer's command or character sets.

With this standard (this is new and not yet implemented on any available Radio Shack printer) codes which are not in a printer's command or character sets will be defined as either an "ignored" code or as an "unprintable" code. If a code is to be ignored (and ignoring a code is a valid response of the printer to an unknown code), the printer will take no action. As far as the printer is concerned the code was never sent. If the code is in the "unprintable" category, the printer will print a special character which will usually look something like an hour-glass (X). The purpose of this character is to tell you that the printer has received a code for a control function or character which is not in the available character set and which is not one of the codes to be ignored.

In the case of our no bell printer and the bell code 7, future printers will simply print the "unprintable code" character (X). In general, printers will ignore Decimal codes 0, 1, 127 and 255. In addition, certain control codes and ESC sequences are sometimes ignored (For instance, if the printer is in the Data Processing mode and you send a DC3 control code, the printer will ignore it). Also, if you send a POS code which moves the print head past the real end of the print line, that code will be ignored.

In Data and Word Processing modes, all unprintable or inactive codes (for that printer) will be printed using the special unprintable code character (X). This includes invalid unprintable codes between 00 Dec and 31 Dec. It also currently includes codes between 128 Dec and 159 Dec and any incorrect repeat sequences. The characters from 192 Dec to 254 Dec will be printed using the unprintable code character when the printer is using proportional space characters. In Bit Image mode, the unprintable code character is never printed. Instead unprintable codes are simply ignored.

Character Sets and Fonts

A printer may have several different character sets available. The common character sets are the 96 character ASCII set, the special character set, and the Block Graphics set. Within each character set (especially ASCII), there may be one or more FONTS. A character font is one particular size and style of character. Among the common character fonts are one or more monospaced (fixed pitch) sets. In a fixed pitch font each character requires the same amount of horizontal space. That is, an I and a W are the same width. The "ordinary" character font for most printers is the standard 10 characters per inch (10 CPI or 10 pitch for print wheels).

In a daisy wheel printer, the pitch is limited to the fixed or proportional characters actually provided on the particular print wheel that you are using. Our daisy wheel printers do allow different print wheels to have different pitches, but you have to supply the proper software for either the WP-50 or the Qume printers to print with the new pitch setting. On the DW II you need only set the switch on the front panel (or send the proper control codes from software) to get 12 pitch spacing.

In a dot matrix printer you will usually have a 5 CPI elongated character font (same height as regular characters, but twice the width), a condensed font, and an elongated condensed font available. This is because dot matrix printers have much more flexibility and can actually manipulate or change the fonts. It is possible to start with a certain character font (7 x 9 for example) and then manipulate that font to provide the varying horizontal pitches we have described, plus, conceivably several others.

A second type of character font is proportionally spaced. In a dot matrix printer these characters will usually have a higher dot matrix print density than fixed pitch characters. This character font will often be described as n by x where x is a fixed number of vertical dots (typically 7 or 9) and n varies from character to character (in an n x 9 set, the characters might vary from 9 to 23 horizontal dots per character with 9 vertical dots in the normal print size). Technology in the printer industry will soon be providing print heads with greater than nine dot vertical density. As these heads become available, watch for exciting new possibilities from dot matrix printers!

The Qume daisy wheel printer is capable of supporting proportional spaced characters, but requires a good deal of software support to accomplish this. The DW II allows either the front panel switch or software to select the proportional space mode. The DW II handles the movement of the carriage properly using internal firmware and therefore requires little software support in order to print proportional characters (Note: right justification can entail considerable software support).

The different character sizes and styles which are available in a dot matrix printer is a function of the printer's design. You have everything that is possible the day you buy the printer. This is not true for a Daisy Wheel printer. With a Daisy Wheel, all you have to do to change character sets or fonts is to buy a new print wheel. We have already announced six print wheels for the Daisy Wheel II, and more will be coming. If you have one of our WP 50 or Qume printers, there are scores of different print wheels available from several sources, including Daisytek Inc. of Richardson, Texas. (800-442-4502 in Texas or 800-527-4212 outside Texas).

The printable characters in each of the printers which fit the new standard may be classified as follows:

Standard 96 Character ASCII

With a few exceptions which we will note below, all Radio Shack printers use the standard 96 character ASCII set.

The most general exception to the 96 character ASCII set comes with regard to decimal values 91-95. Some of our printers print arrows for the standard ASCII characters, and three printers have both sets of characters available (LP IIIA, LP V and LP VI). If you want the alternate characters, you will need to have your

(Continued on Page 13)

printer changed by a Radio Shack Repair Facility. The following chart lists both the standard ASCII characters and the arrow characters. For each printer, if character is available we have given you the Decimal value of the character. If the character is not available we put a '—."

ASCII Character	LP I	II	IIA	III	IIIA	IV	V	VI	VII	WP50	QUME	DWII	QP	QPII	PLOT
[	91	91	91	91	R	91	R	91	91	91	91	91	91	—	—
\	92	92	92	92	R	92	R	92	92	92	—	92	92	—	—
]	93	93	93	93	R	93	R	93	93	93	93	93	93	—	—
^	94	—	—	—	—	94	—	—	94	94	—	94	94	—	—
_	95	—	—	R#	95*	95	95*	R#	95	95	—	95	95	95	—
Arrow Characters															
↑	—	94	94	R#	91*	—	91*	R#	—	—	—	—	—	91	—
→	—	96	96	R	92	—	92	R	—	—	—	—	—	92	—
↓	—	95	95	R#	93*	—	93*	R#	—	—	—	—	—	93	—
←	—	126	126	R	94	—	94	R	—	—	—	—	—	94	—

R— Available requires Repair Facility to make change
*Available in the alternate character set (UP) — 94, (LF) — 95. In the regular set, decimal 95 is the underscore character
#Available in regular character set (UP) — 94, (LF) — 95. In the alternate set, decimal 95 is the underscore character

The next difference comes because four of our printers use a seven-bit parallel interface instead of an eight-bit parallel interface. Since the eighth (most significant) bit is missing, the Line Printer I (26-1150 and 26-1152), the Quick Printer (26-1153), the WP 50 (26-1157) and the Qume Daisy Wheel Printer (26-1157A) repeat their character sets for decimal values 128 through 255. This makes it look like you have two complete character sets (control codes and all) the first from decimal 0 through 127 and the second identical set from 128 to 255. Note: The Line Printer II is an eight bit printer, but uses the eighth bit to designate elongated characters. The result is two not quite identical sets of characters, similar to the seven bit printers.

The Line Printer I (26-1150 and 26-1152) and the Plotter/Printer both have Upper Case Alphabetic characters only. The Line Printer I translates lower case characters to upper case, while the Plotter/Printer ignores lower case characters completely.

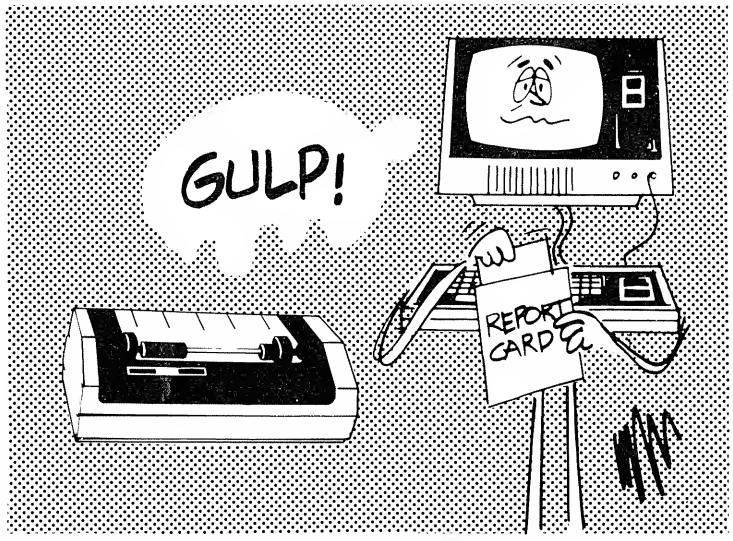
The last set of variations to the 96 character ASCII set come from the Qume printer. The following chart shows all of the discrepancies between standard 96 Character ASCII and the standard 96 Character Qume Word Processing printwheel (other series of Qume printwheels have different discrepancies):

DEC.	ASCII Symbol	QUME Symbol
92	\	®
94	^	©
96	`	°
123	{	§
124	|	¶
125	}	†
126	~	™

Conclusion

What we have described in this later portion of this article is a new set of printer character standards. You should remember that these standards are new (and we hope not subject to very much change, other than the addition of provisions for new features) and not fully implemented on any printer. We have also tried to show you how our current and past printers meet these standards. We have not described every possible function on all existing printers. Check your printer manual for more information on how your printer responds to codes, and for possible additional capabilities which your printer may have.

After hearing about this article and what it purposed to do, one of our buyers described the task as being an Adventure game. That is, you are never sure that you have found all of the treasure or met all of the demons. We think we have covered all of the possibilities and made provision for the future. If we are wrong, the standards will be modified and we will let you know about those changes.



Pocket Computer (From Page 21)

During the past week I have had several phone calls from Pocket Computer owners who suggested a method to help you avoid forgetting a PC program name which you have stored on tape. Prior to recording your program, use the built-in microphone in your tape recorder to record verbally the name of the program (spell it out phonetically so there won't be any doubt) and a short description of what it does. Now stop the recorder and reconnect it to the interface and proceed with the normal CSAVE function . . . I record it twice, one to load and the second to verify. If you do this before each program you save on a tape, you can locate a particular program easily by using the CUE feature on most recorders to find your voice description. I have tried this and it works quite well . . . the computer simply ignores the voice signals on the tape when trying to read in a program.

Before closing for this month, here's a little feature of the Pocket Computer which relates to the INPUT and AREAD statements (see OM pages 52 and 53, 69 and 70 respectively) when using a numeric variable. When input to a variable is from the keyboard, a number of things may happen before the variable is actually loaded. If you enter a single number (one or more digits) into the display, it of course will go directly into the variable. However, if you enter a mathematical expression such as 5*8, the math is done first and the result (40 in this case) is put into the specified variable of the INPUT or AREAD statement. This is mentioned in the illustration at the top of page 53. It is also possible to use another numeric variable as input by typing its letter on the keyboard, and the contents of the memory location will be loaded into the specified variable of your INPUT or AREAD statement. Here again, you can combine this feature with math functions, such as P * 2.5, as input to a numeric variable. If P contained 10 then the variable in your statement would end up with 25.

Well, I hope this month's tidbits of information prove useful to you as you explore new tasks to program on your Pocket Computer . . . so until next month . . . more Pocket Power to you!

P.S. Everyone who has purchased a TRS-80 Pocket Computer is entitled to receive a one year FREE subscription to this newsletter. So, if you know someone who isn't getting it, tell them to send a copy of their sales receipt or other proof of purchase along with a short note to the Editor at P.O. Box 2910, Fort Worth, TX 76101. We'll see that they get on the mailing list and I'm sure they will appreciate it.

Model I/III Bugs, Errors, and Fixes

Model III TRSDOS Manual (26-1063)

Page 102 of the Model III TRSDOS Manual lists a 300 Baud terminal program. An error in line 65 of the program causes output characters to be echoed to the screen, but not through the RS-232 to the host computer. Change line 65 of the program to read:

```
65 DEFUSR2=&H0055
```

Model III BASIC

There are some restrictions on using BASIC in a DO file. If BASIC is part of your DO file, you can answer "How many files?" and "Memory size?" inside your DO file. In addition, after these have been answered, you can have RUN "filename" or LOAD "filename" (but not both) as part of your DO file. However, if one of these is used, it would have to be the last item in the DO file.

EXAMPLE:

```
BUILD TEST (ENTER)
Hit BREAK to exit
Type in up to 63 characters
BASIC (ENTER)
Type in up to 63 characters
(ENTER)
Type in up to 63 characters
(ENTER)
Type in up to 63 characters
RUN "PROG/BAS" (ENTER)
Type in up to 63 characters
(BREAK)
TRSDOS Ready
DO TEST (ENTER)
```

Another application for a DO file with BASIC could be to write a program within the DO file for execution under BASIC (Do not use AUTO line number).

EXAMPLE:

```
BUILD TEST2 (ENTER)
Hit BREAK to exit
Type in up to 63 characters
BASIC (ENTER)
Type in up to 63 characters
10 FOR X=1 TO 10 (ENTER)
Type in up to 63 characters
20 PRINT X: NEXT X (ENTER)
Type in up to 63 characters
RUN (ENTER)
Type in up to 63 characters
(BREAK)
TRSDOS Ready
DO TEST2 (ENTER)
```

Again, RUN would have to be the last command in the DO file. LIST could be substituted for RUN, but then LIST would have to be the last command. (You could not use both.)

These restrictions are due to the fact that once you get into BASIC, BASIC will continually scan the keyboard for the (BREAK) key character. During this scan, any other characters will be thrown away. This is because the DO file does not know that BASIC is scanning the keyboard and in turn BASIC will "drain" the DO file. Instead of the keyboard supplying the key input, characters are supplied by the DO file.

Tape Payroll (26-1504)

Version 1.2 of Tape Payroll gives erratic line-feeds when printing out checks or reports with newer printers.

Follow the following procedure to correct this problem:

1. CLOAD the PRPROC program into memory.
2. At READY, type LIST 2130 and press (ENTER).
3. Look at the line until you find CHR$(138).
4. At READY, type EDIT 2130 and press (ENTER).
5. Press the (SPACEBAR) until the cursor is on top of the C in CHRS$(138). Type (9) (D), press the (I) key once, and type in (") (SPACEBAR) (") then press (ENTER).
6. Follow steps 2 through 5 and replace the CHR$(138)s in lines 2170, 2190, 3250 and 3900.
7. CSAVE the program onto a fresh tape and use this new tape from now on.
8. CLOAD the PRTOT program into memory.
9. At READY, type LIST 1080 and press (ENTER).
10. Look at the line until you find CHR$(138).
11. At READY type EDIT 1080 and press (ENTER).
12. Press the (SPACEBAR) until the cursor is on top of the C in CHR$(138). Type (9) (D), press the (I) key once, and type (") (SPACEBAR) (") then press (ENTER).
13. Follow steps 9 through 12 and replace CHR$(138) in lines 1160 and 1180.
14. CSAVE the program onto a fresh tape and use this tape from now on.

Inventory Control System (26-1553)

For Model III only, in version 1.1 of ICS if the prompt "Is the Date Set Correctly" is answered with N for no, and you then give the correct date, the machine hangs up and RESET must be pushed.

To solve this problem, make the following changes to the ICS program:

1. At TRSDOS Ready, type BASIC and press (ENTER).
2. Answer the prompts as usual. At READY, type LOAD "ICS" and press (ENTER).
3. At READY, type LIST 160 and press (ENTER).

The screen will show:

```
160 POKE 16454,VAL(LEFT$(DT$,2)): POKE 16452,
    VAL(RIGHT$(DT$,2))
```

4. At READY, type EDIT 160 and press (ENTER).
5. Press the (SPACEBAR) until you are on top of the first 4 in 16454. Press (C) (9) (C) (2) (C) (4). Now press the (SPACEBAR) until you are on top of the 4 in 16452. Press (C) (9) (C) (2) (C) (2), and press (ENTER).
6. At READY, type LIST 160 and press (ENTER). The line should now read:

```
160 POKE 16924,VAL(LEFT$(DT$,2)): POKE 16922,
    VAL(RIGHT$(DT$,2))
```

7. At Ready, type LIST 170 and press (ENTER).
8. The screen will show:

```
170 POKE 16453,VAL(MID$(DT$,4,2))
```

9. At Ready, type EDIT 170 and press (ENTER).
10. Press the (SPACEBAR) until the cursor is on top of the 4 in 16453, then press (C) (9) (C) (2) (C) (3) and press (ENTER).
11. At Ready, type LIST 170 and press (ENTER).
12. The screen should show:

```
170 POKE 16923,VAL(MID$(DT$,4,2))
```

13. To save your changes, type SAVE"ICS" and press (ENTER).

Disk Payroll (26-1556)

The non-EIC version of both Model I and Model III Disk Payroll has erratic line feeds when printing checks with Line Printer VI

(Continued on Page 15)

Disk Payroll (From Page 14)

and the LPC driver. This occurs especially during check printing and may also occur with the Daisy Wheel II and Line Printer V.

To solve this problem, we suggest that you get the most recent EIC version of Disk Payroll. This is available through your Radio Shack store as stock number 700-3003 and is available for a suggested retail price of $50.00 with proof of purchase for 26-1556.

If you do not wish to get the EIC version, make the following changes to the PR4CHK program:

1. At TRSDOS READY, type in BASIC and press (ENTER). Answer the two prompts as usual.
2. When READY appears on the screen, type LOAD "PR4CHK" and press (ENTER).
3. When READY appears again, type LIST 380 and press (ENTER).
4. The screen will show line 380. In line 380, locate the phrase CHR$(138).
5. At READY, type EDIT 380 and press (ENTER). Now press the (SPACEBAR) until your cursor is on top of the (C) in CHR$(138). Press (9) (D). Now, press the (I) key once and then (") (SPACEBAR) (") to get " " and press (ENTER). Your line is now correct.
6. Follow steps 3 through 5 and change all of the CHR$(138)s in lines 440, 460, 820, 840, 900, 920, 940, 5030, 5060, 5100, 5180, 5200, 8000, 8015, and 8045.
7. Replace lines 500, 980, 5220 and 8060 as follows:

```
500 FORQL=1TO10: LPRINT" ": NEXT (ENTER)
980 FORQL=1TO6: LPRINT" ": NEXT (ENTER)
5220 FORQL=1TO6: LPRINT" ": NEXT (ENTER)
8060 FORQL=1TO10: LPRINT" ": NEXT (ENTER)
```

8. Check each of the changes carefully to be sure it matches what we have here. To save the changes you have made, type SAVE"PR4CHK" and press (ENTER).

Profile (26-1562)

On certain newer printers, Profile version 3.0 on both Models I and III is generating extra linefeeds when it does a Top-of-Form.

To correct this in the **Model III** version, follow this step:

1. At TRSDOS Ready, enter the following command:

```
PATCH PRINT (ADD=79EE,FIND=8A,CHG=20) (ENTER)
```

This will permanently correct the Model III version.

To correct the **Model I** version, follow this procedure:

1. Insert a TRSDOS diskette other than your Profile diskette into the machine.
2. At TRSDOS READY, type BASIC and press (ENTER).
3. At READY, after you have answered the start-up prompts, enter the following BASIC program:

```
10 REM "PROFIX"
20 CLEAR 1000
30 OPEN "R",1,"INIT"
40 FIELD 1, 100 AS FD$
50 GET 1,1
60 CH$=FD$
70 MID$(CH$,76,1)=CHR$(66)
80 MID$(CH$,81,1)=CHR$(0)
90 LSET FD$=CH$
100 PUT 1,1
110 CLOSE
120 OPEN "R",1,"PRINT"
130 FIELD 1, 255 AS FD$
140 GET 1,7
150 CH$=FD$
160 MID$(CH$,255,1)=CHR$(32)
170 LSET FD$=CH$
180 PUT 1,7
190 CLOSE
```

4. Now type SAVE"PROFIX" and press (ENTER).
5. At READY, remove the TRSDOS diskette and place the Profile program diskette in drive 0.
6. Type RUN and press (ENTER).
7. The program PROFIX will change the Profile programs to correct the problem. When the screen shows READY, the changes are complete.
8. Press the RESET button and run Profile as instructed in the manual.

Scripsit™ (26-1563)

In both Models I and III, Scripsit versions 1.0 and 3.0, the Widow suppress mode does not work with Double Line spacing.

This problem can not be corrected in the current versions, but can be avoided by using an odd number of lines in the Top Margin.

For instance, when the format line reads:

```
>TM=6 RM=72 LS=2 WS=Y
```

The widow suppress will not work, but if the line reads:

```
>TM=5 RM=72 LS=2 WS=Y
```

widow suppress will work correctly.

VisiCalc™ (26-1566)

If you wish to print your VisiCalc screens using Scripsit, use the / P F format in VisiCalc. This will save the sheet in Printing Format, enabling you to load the sheet into Scripsit by using filename/PRF.

If you have been having problems printing VisiCalc on a Serial printer, but your Scripsit works alright with the serial printer, this method may also allow you to print VisiCalc files by going through Scripsit.

T-Bug (26-2001)

The T-Bug manual contains a misprint on page 29:

```
DDG05 LD H,(IX+IND)
```

should read:

```
DDG6G05 LD H,(IX+IND)
```

Model II Bugs, Errors, and Fixes

Inventory Management System (26-4502)

There are problems printing purchase orders using the Line Printer VI. The problem is that random line-feeds and extra lines are appearing. This same problem may also occur with the Daisy Wheel II, Qume, and the Line Printer V, and similar problems can be expected with future new printers.

To solve this problem, make the following changes to the ORDER/BAS program:

1. At TRSDOS READY, type BASIC and press (ENTER)
2. At Ready, type LOAD"ORDER/BAS" (ENTER)
3. At Ready, type LIST 5300 (ENTER)
4. The screen will show:

```
5300 FOR X=1TOLF:LPRINTCHR$(138):NEXT:RETURN
```

5. At Ready, type EDIT 5300 (ENTER)
6. Press (S) (C). The cursor should now be on the C of the CHR$(138). Press (9) (D). Press (I) once, then press (") (SPACEBAR) (") to get " " then press (ENTER)
7. At Ready, type LIST 5300 (ENTER) (Continued on Page 16)

Inventory Management (From Page 15)

8. The screen will show:

```
5300 FOR X=1TOLF:LPRINT" ":NEXT:RETURN
```

9. At Ready, type LIST 5170 (ENTER).
10. The screen will show:

```
5170 LF=56-ML-LC:GOSUB5300:LPRINTLL$
```

11. At Ready, type EDIT 5170 (ENTER).
12. Press the (SPACEBAR) 5 times, press (C) (5), and press (ENTER).
13. At Ready, type LIST 5170 (ENTER)
14. The screen will now show:

```
5170 LF=55-ML-LC:GOSUB5300:LPRINTLL$
```

15. At Ready, type SAVE"ORDER/BAS" (ENTER).
16. At Ready, type SYSTEM (ENTER).
17. The line changes are now complete and saved. You may now continue normal operation.

Payroll (26-4503)

The list of employees prints all names except those in which the last name ends with the letter "Z."

To correct this problem, in line 1740 of the "ADD" program, change the upper case "Z" to a lower case "z." The corrected line should read as follows:

```
1740 FORI=1TO200:MID$(C1$,I)=CHR$(I):
     H$(I)=STRING$(30,"z"):NEXTI:IFLOF(3)<1THEN1800
     ELSEFORN=1TOLOF(3):GET3,N:PRINT@(9,25),
     MID$(F3$,1,30);:MID$(H$(N),1,30)=MID$(F3$,1,30):
     NEXT:N=0
```

After the line has been corrected, and a copy of the program saved, change or add to employee information to force the program to resort the data.

In Payroll versions 1.0 and 1.1 State Unemployment tax is limited to two decimal places. Some states need three decimal places for this rate.

To allow three decimal places, change the FL=3 in lines 445 and 760 of the INPUT program to read FL=6. Be sure you save a copy of the corrected INPUT program.

Accounts Receivable (26-4504)

Users of Accounts Receivable versions 1.0 and 1.1 are getting ERROR 94s in different places. There are two known reasons for getting an Error 94 in a data file.

1. Right after setup the user has defined the system capacity as 2 or 3 drives, and did not have blank formatted diskettes in drives 2 and/or 3.

2. During normal operations (e.g. print function, posting, invoices) the user hits the reset key or ends the job abnormally.

If the error occurs because of condition 1, the only answer is to setup all over again, making sure the system capacity and the number of drives you desire to use are equal, and that you have blank formatted diskettes in the appropriate drives.

If the error occurs because of condition two, first try:

1. Use TRSDOS to LIST all nine of the data files.

2. If an Error six is encountered, list that data file until the EOF appears. If this does not cure the problem, use a good backup and re-enter any data since that backup was made.

3. If an Error 31 is encountered when trying to list a file, look in all directories for the first file listed when you got the Error 94. If the file is not found, the file has been lost and you should go to a good backup.

Accounts Payable (26-4505)

When updating the Accounts Payable to the General Ledger, an error message "Total Entries exceeds 50. G/L TRANSFER ABORTED" will be received if the number of Account Payable entries exceeds 50.

To solve this problem, change line 190 of the "APGLPOST/BAS" program to read:

```
190 A=A+1:A(A)=GA:V#(A)=GC#:GOSUB240:IFA=1THENRETURN
    ELSEIFNE<24THENPUT2:A=0:A(2)=0:V#(2)=0D0:NE=NE+1:
    RETURN
```

Then add line 195:

```
195 NX=0:V#(2)=-T#:T#=-GC#:A(2)=G(1):GOSUB240:PUT2:
    GOSUB230:G=G-1:A=0:A(2)=0:V#(2)=0D0:NE=0:NX=1:
    RETURN
```

Be sure to save a copy of the corrected program.

When selecting invoices by Discount/Due date, using version 1.0, ALL invoices are selected. Also after adding 500 vendors, customers have tried to delete 10 then re-add 10 vendors and could not. Instead they received a "Cannot add to Vendor List, Vendor's List Full" error message.

To correct these problems, use the following steps:

1. At TRSDOS Ready, type BASIC (ENTER)
2. At Ready, type LOAD"APS/BAS" (ENTER)
3. At Ready, type LIST 12212 (ENTER)
4. The screen should show:

```
12212 GET1,K:IFDU<>0ANDDD<>0ANDABS(CVI(I3$))<=DUAND
      CVI(I4$)<=DDTHENGOSUB12300
```

5. At Ready, type EDIT 12212 (ENTER)
6. The screen will show 12212, press (2) (S) (T). The cursor should be on the T of THENGOSUB12300. Press (I) once and type the following:

```
ANDABS(CVI(I3$))>0ANDABS(CVI(I4$))>0 (ENTER)
```

7. At Ready, type LIST 12212 (ENTER)
8. The screen should show:

```
12212 GET1,K:IFDU<>0ANDDD<>0ANDABS(CVI(I3$))<=DUAND
      CVI(I4$)<=DDANDABS(CVI(I3$))>0AND
      ABS(CVI(I4$))>0THENGOSUB12300
```

9. At Ready, type LIST 12225 (ENTER)
10. The screen will show:

```
12225 IF DU=0ANDABS(CVI(I4$))<=DDTHENGOSUB12300
```

11. At Ready, type EDIT 12225 (ENTER)
12. The screen will show 12225. Press (1) (S) (T). The cursor should stop on top of the T in THENGOSUB12300. Press the (I) key once and type the following:

```
ANDABS(CVI(I3$))>0ANDABS(CVI(I4$))>0 (ENTER)
```

13. At Ready, type LIST 12225 (ENTER)
14. The screen will show:

```
12225 IF DU=0ANDABS(CVI(I4$))<=DDAND
      ABS(CVI(I3$))>0ANDABS(CVI(I4$))>0THENGOSUB12300
```

15. At Ready, type LIST 3540 (ENTER)
16. The screen should now show:

```
3540 GET1,K:GOSUB2030:IFK=VJTHEN1800ELSEK=CVI(IJ$):
     GOTO3540
```

17. At Ready, type EDIT 3540 (ENTER). Press (2) (S) (1). The cursor should be on top of the 1 in 1800ELSE. Press (4) (D) then (I) once. Type:

```
RETURN (ENTER)
```

18. At Ready, type LIST 3540 (ENTER)
19. The screen should now show:

```
3540 GET1,K:GOSUB2030:IFK=VJTHENRETURNELSEK=CVI(IJ$):
     GOTO3540
```

20. At Ready, type SAVE"APS/BAS" (ENTER)
21. At Ready, type SYSTEM (ENTER)
22. The line changes to APS/BAS are now complete.

In version 1.0 there is also a problem in the APCHECKS/BAS program. The program now points to "Align Checks" after you

(Continued on Page 17)

Accounts Payable (From Page 16)

have updated vendors. The following changes will modify the program to point to "Return to Main Menu":

1. If you just completed the changes to APS/BAS you should still be in BASIC. If you are not in BASIC, follow step 1 above to load BASIC.
2. At Ready, type LOAD"APCHECKS/BAS" (ENTER)
3. At Ready, type LIST 4002 (ENTER)
4. The screen will now show:
   ```
   4002 IFEP=-3THENEP=5:AM=1ELSEAM=0:EP=5
   ```
5. At Ready, type EDIT 4002 (ENTER)
6. Press (S) (5). The cursor should now be on top of the 5 in the first EP=5. Press (C) (4) then (S) (5). The cursor should now be on top of the 5 in the second EP=5. Press (C) (4) (ENTER).
7. At Ready, type LIST 4002 (ENTER)
8. The screen will now show:
   ```
   4002 IFEP=-3THENEP=4:AM=1ELSEAM=0:EP=4
   ```
9. At Ready, type LIST 7002 (ENTER)
10. The screen will now show:
    ```
    7002 IFABS(EP)=10REP=5THENEP=0
    ```
11. At Ready, type EDIT 7002 (ENTER)
12. Press (S) (5). The cursor should now be on top of the 5 in EP=5. Press (C) (4) (ENTER).
13. At Ready, type LIST 7002 (ENTER)
14. The screen will now show:
    ```
    7002 IFABS(EP)=10REP=4THENEP=0
    ```
15. At Ready, type SAVE"APCHECKS/BAS" (ENTER)
16. At Ready, type SYSTEM (ENTER)
17. The line changes APS are now complete, and you can continue normal operation of the program.

Profile II (26-4512)

In version 1.0 of Profile II, after setting up a screen format the operator will be unable to add any information to fields 01 through 09. These fields are shown on the screen but will not allow any input.

If you will go back to the screen format and change the fields from 01 to 1, 02 to 2, etc., this will solve the problem (the leading zero is causing the problem).

On more than one occasion a Profile user has set-up Profile (version 1.0) on a single disk, only to find later that the number of records requires more than one drive. The following directions explain how to expand your Profiles II from 1 drive to several. In Profiles it is possible to COPY data files from one drive to another and then kill the file where you don't want it without any problems. Profile II will find the file wherever it is.

Use the appropriate procedure, depending on the number of segments you have in your Profile II data file, and the number of drives you wish to use. In each of the following, the *** should be replaced by the file name you are using for your data file, padded on the right with enough zeros to make eight characters.

TWO SEGMENTS—ONE EXPANSION DRIVE

If you have two segments and 1 expansion drive the following should be copied from drive 0 to drive 1 to allow for maximum capacity. After these files have been copied to drive 1 KILL them off Drive 0. MAKE A BACKUP BEFORE YOU BEGIN.

***/DAT				
***/PMP	***/PM2	***/PM3	***/PM4	***/PM5
***/PRT	***/PR2	***/PR3	***/PR4	***/PR5
***/LB1	***/LB2	***/LB3	***/LB4	***/LB5

THREE SEGMENTS—ONE EXPANSION DRIVE

If you have three segments and 1 expansion drive the following should be copied to drive 1 to allow for maximum capacity. After these files have been copied to Drive 1 KILL them off Drive 0. MAKE A BACKUP BEFORE YOU BEGIN.

***/DAT	***/DA2			
***/PMP	***/PM2	***/PM3	***/PM4	***/PM5

FOUR SEGMENTS—ONE EXPANSION DRIVE

If you have four segments and 1 expansion drive the following should be copied to drive 1 to allow for maximum capacity. After these files have been copied to Drive 1 KILL them off Drive 0. MAKE A BACKUP BEFORE YOU BEGIN.

***/DA2	***/DA3			
***/PMP	***/PM2	***/PM3	***/PM4	***/PM5
***/PRT	***/PR2	***/PR3	***/PR4	***/PR5
***/LB1	***/LB2	***/LB3	***/LB4	***/LB5

TWO SEGMENTS—TWO EXPANSION DRIVES

It you have two segments and 2 expansion drives the following should be copied to drives 1 & 2 to allow for maximum capacity. After these files have been copied to Drive 1 and 2 KILL them off Drive 0. MAKE A BACKUP BEFORE YOU BEGIN.

Drive 1 ***/KEY
Drive 2 ***/DAT

THREE SEGMENTS—THREE EXPANSION DRIVES

If you have three segments and 3 expansion drives the following should be copied to drives 1, 2, & 3 to allow maximum capacity. After these files have been copied to Drive 1, 2, and 3 KILL them off Drive 0. MAKE A BACKUP BEFORE YOU BEGIN.

Drive 1

***/DAT				
***/PMP	***/PM2	***/PM3	***/PM4	***/PM5

Drive 2

***/DA2				
***/PRT	***/PR2	***/PR3	***/PR4	***/PR5

Drive 3

***/DA3				
***/LB1	***/LB2	***/LB3	***/LB4	***/LB5

FOUR SEGMENTS—TWO EXPANSION DRIVES

If you have four segments and 2 expansion drives the following should be copied to drives 1 and 2 to allow for maximum capacity. After these files have been copied to Drive 1 and 2 KILL them off Drive 0. MAKE A BACKUP BEFORE YOU BEGIN.

Drive 1

***/DAT				
***/PMP	***/PM2	***/PM3	***/PM4	***/PM5
***/PRT	***/PR2	***/PR3	***/PR4	***/PR5
***/LB1	***/LB2	***/LB3	***/LB4	***/LB5

Drive 2

***/DA2	***/DA3

FOUR SEGMENTS—THREE EXPANSION DRIVES

If you have four segments and 3 expansion drives the following should be copied to drives 1, 2, and 3 to allow for maximum capacity. After these files have been copied to Drive 1, 2, and 3 KILL them off Drive 0. MAKE A BACKUP BEFORE YOU BEGIN.

Drive 1

***/DAT				
***/PMP	***/PM2	***/PM3	***/PM4	***/PM5

Drive 2

***/DA2				
***/PRT	***/PR2	***/PR3	***/PR4	***/PR5

Drive 3

***/DA3				
***/LB1	***/LB2	***/LB3	***/LB4	***/LB5

NOTE: THE *** REPRESENTS YOUR FILE NAME.

Scripsit™ 26-4530

Two problems have been identified in Scripsit when you use horizontal format files.

The first is an error in storing tabs in both 1.0 and 1.0a. When a tab stop is set on certain columns, the cursor will disappear and the system will lock-up. This forces you to RESET the computer. This does not damage anything on the disk! However, any text that had not been written to the disk is lost.

The second problem will correct problems in 1.0 only in which text is lost during printout even though it looks fine on the video and/or Print prompts appear garbled.

To solve these problems, follow this procedure:

1) RESET your system and insert a Scripsit diskette.
2) Answer the DATE prompt.
3) Press the (HOLD) key.
4) Answer the TIME prompt (or press (ENTER)).
5) At TRSDOS Ready, carefully enter the following patches:

To correct the tab problem (1.0 and 1.0a):

```
PATCH SCRIPSIT A=9C9E F=78C621 C=CD30DC
PATCH SCRIPSIT A=9CB2 F=2802F680D621 C=CD40DC000000
PATCH SCRIPSIT A=DC30 F=00000000000000000000000
                      C=78FE5F3003C621C9C642C9
PATCH SCRIPSIT A=DC40 F=00000000000000000000000
                      C=2805F680D642C9D621C9
```

To correct the loss of text problems (1.0 only):

```
PATCH SCRIPSIT A=C96B F=CDCBC610FB C=CD10DB0000
PATCH SCRIPSIT A=DB10 F=000000000000 C=F5783DFD862C
PATCH SCRIPSIT A=DB16 F=00000000000000000
                      C=322F90783DFD8637
PATCH SCRIPSIT A=DB1E F=00000000000000000
                      C=323A90F148C3CDC6

PATCH SCRIPSIT A=C6D0 F=CDE4C6 C=C330DB
PATCH SCRIPSIT A=DB30 F=000000000000 C=F5C5E52B2BBE
PATCH SCRIPSIT A=DB38 F=00000000000000000000
                      C=C24ADB2BBE23CA4ADB
PATCH SCRIPSIT A=DB41 F=00000000000000000000
                      C=237E8177E1C1C353DB
PATCH SCRIPSIT A=DB4A F=00000000000000000000
                      C=E1C1F1CDE4C6C3D3C6
PATCH SCRIPSIT A=DB53 F=000000000000 C=FD352DC3ECC6
```

6) After making these patches, type STARTUP and press (ENTER). Scripsit will load.
7) BACKUP the patched Scripsit disk!

Another problem which is occurring in both version 1.0 and 1.0a is that the printer is not pausing between pages. This happens particularly when merging documents. If you have already checked the print menu and made sure there is a Y for the question "PAUSE BETWEEN PAGES?" then insert the following patches

Solution:

1. Reset the system at SWAP diskettes
2. Answer the Date Prompt
3. Press (HOLD) and press (ENTER) for the Time Prompt
4. Type in the following patches at TRSDOS Ready:

```
PATCH SCRIPSIT A=C40F F=FD210390 C=C380DB00
PATCH SCRIPSIT A=DB80 F=0000000000000000
                      C=E521FFFF2242BDE1
PATCH SCRIPSIT A=DB88 F=00000000000000
                      C=FD210390C313C4
```

5. Type STARTUP and press (ENTER)
6. Scripsit is now patched and you can continue operation.

COBOL Compiler (26-4703)

There are two areas of possible interest for COBOL users.

Binary data in ISAM key fields causes loop in COBOL. This is an unusual programming practice but can be allowed with the following patches. Note: do not make this PATCH unless you are sure you know how to handle binary data in ISAM key fields.

Put in the following patches at TRSDOS Ready pressing (ENTER) after each patch.

```
PATCH RUNCOBOL A=502A F=FA31503E01 C=CDE0830000
PATCH RUNCOBOL A=83E0 F=0000000000000000
                      C=2B963E01D03E04C9
```

Problem:

Executing RUNCOBOL from a DO file allows operator the freedom to press BREAK at any time. If ISAM file indexes are being updated, pressing BREAK will destroy the file so that it is unusable. To disallow this possibility, use the following patch:

Solution:

```
PATCH RUNCOBOL A=7077 F=CD C=CC
```

This patch prevents RUNCOBOL from being executed in a DO file.

(Continued on Page 20)

New Customer Service Phone Lines

Customer Service has added people and telephone lines to better support our customers. To facilitate handling of calls, we now have separate phone numbers for Model I/III Business Software, and for Model II Business Software. Here are the numbers:

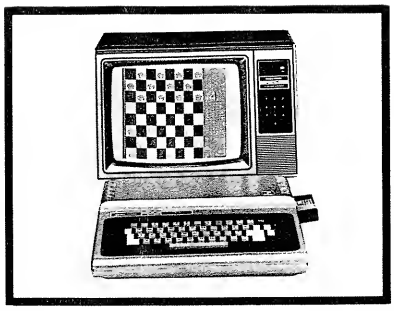
Model I/III Business Software	1-800-433-5641
In Texas	1-800-772-5973*
Model II Business Software	1-800-433-5640
In Texas	1-800-772-5972*
All Other Calls	1-800-433-1679
In Texas	1-800-772-5914

*These Texas WATS lines will not be available until May 20, 1981. Until that time, Texas Business Software users should continue to use the 800-772-5914 number.

When you call Customer Services, we will be able to help you better if you have the following information readily available:

1) Your Name.
2) Your phone number, where we can reach you.
3) What TRS-80 Computer System you are using and the number of disk drives, if applicable.
4) The name and stock number of the software package you are using. If you know the version number, we would also like that information.
5) What Error Codes you have received.
6) How the error occurred, and the function you were executing when it occurred.
7) Information about any patches or program corrections you may have made.

These items are particularly important if you are calling one of the four new Business Software numbers. These new numbers have been set up so that your call will be answered by a Customer Service Representative and not a receptionist. The two All Other Calls lines are the old WATS numbers, and will continue to be answered by a receptionist who can route the call to the most qualified available Customer Service Representative. We are trying to handle a large volume of calls and any help you can give us is appreciated.



Color Computer

Product Line Manager's News

Remember the "little" house we built last month using the LINE command? Well, hold onto your hat, 'cause we're going to do it again this month using a couple of other commands in the arsenal of Extended Basic. But, before we proceed, Mr. George A. Burns of Ozark, MO has brought to our attention a small misprint in the PINBALL (26-3052) manual. After editing the playing field, the manual says you can press P to begin playing your creation. What, in fact, you must do is first press the space bar to get back to the Edit menu, then select P to play. Our thanks to you George and future reprints will be corrected. Now, on with the house-building.

As in previous articles on Extended Basic, we must first set up our program to go into the graphics mode:

```
1 DRAW"S4"
5 CLS
10 PCLEAR 4
15 PMODE 3,1
17 PCLS
20 SCREEN 1,0
```

As you might have noticed, we have added another instruction (line #1) to our setup routine. (More on this a little later. For now, just type it in.) Again, for those who might have missed last month's issue or who are proud, new owners, this is what we have done:

Line 5 clears the text screen.
Line 10 clears 4 "pages" of memory for graphics.
Line 15 selects resolution mode 3 (128x192 with 4 colors), starting the display on page number 1.
Line 17 clears the first page of graphic memory.
Line 20 switches the video to the graphics mode and selects one of the two color sets available. (In this case: red, yellow, green, blue.)

Back to that funny DRAW statement . . .

The syntax in programming Extended Color Basic for the DRAW command is as follows:

DRAW line — where "line" can be a string constant or a substring. You can specify 8 different directions and distance, the color you want drawn, change the angle, or change the scale to which it is drawn. You can also move the cursor without drawing a line or draw a blank line (using the background color) or draw the line without updating the position of the cursor.

The directions are entered using the following keys to move various directions:

M = Move the cursor position
U = Up
D = Down
L = Left
R = Right
E = 45 degree angle
F = 135 degree angle
G = 225 degree angle
H = 315 degree angle
X = execute a substring and return
C = Color (1-8 depending on graphics mode selected)
A = Angle (0-3 rotates command & subsequent commands:
    0 = 0 degrees
    1 = 90 degrees
    2 = 180 degrees
    3 = 270 degrees
S = Scale (1-62 based on ¼ths)
    S4 = full scale (4/4)
N = No update of cursor position
B = Blank (no draw, just move)

We'll also be using the PAINT command in this version of the program. Its syntax in Extended Basic is:

PAINT (x,y),c,b

"x" and "y" are coordinates on the screen with x being 0-255 and y being 0-191. "c" is the color you wish to PAINT with and b is the color you wish to PAINT to. If, while PAINTing, the computer meets a color other than the one it is to PAINT to, it will PAINT right over it.

These instructions might sound a little bit awesome at first, but as we write our program, they should become clearer. So . . .

We'll proceed with the program we started at the beginning of this article. Enter these lines:

```
30 DRAW"C4;BM0,185;R255"
35 DRAW"BU70;L25;BL180;L50"
```

A quick explanation of line #1 first: we have specified S4 or full scale drawing. If we left this line out, the computer would have defaulted to full scale anyway. It will stay in this scale mode until we tell it to change to another scale.

In line 30, we're telling the computer to select color 4 (C4; red, in this color set) to draw the line with. Next in line 30 we tell the computer to move the cursor (M) using a blank line (B ;no draw, just move) to position 0,185. Finally in line 30, we're saying to move the cursor right 255 spaces (R255). The punctuation is defined as: the comma between the x and y coordinates is required, but the semi-colons between the directions is not. (They make it easier to read the program when keying in or editing your program.)

In line 35, we're instructing the computer to DRAW a blank line up 70 spaces (BU70). Since we did not specify a starting position in this DRAW command, the computer uses the current cursor position to start from. If we would have started the program with this statement, the computer would have started DRAWing the line from the center of the screen. Then we tell it to move left 25 spaces drawing a line (L25). Then move 180 spaces to the left without drawing a line (BL180). And finally, continue drawing the line 50 more spaces to the left (L50).

Since these instructions are string constants (they don't change), they must be enclosed in quotes.

Enter lines 40-85:

```
40 DRAW"BR50;D20;R150;U30;L50;BD30;U50;L100;D50"
50 DRAW"BU50;E15;R100;G15"
55 DRAW"E15;F15;D5"
60 DRAW"BL30;BD15;E15;R50;G15"
65 DRAW"E15;F15;D23"
70 DRAW"BL30;BD6": FOR X=1 TO 7: DRAW"R4;U1": NEXT X
80 DRAW"BL74;BD7;U20;R40;D20;U5;L40;U5;R40;U5;L40"
85 PAINT(60,80),2,4: PAINT(155,103),2,4
```

In lines 40-80 we are telling the computer to move the cursor around the screen leaving a trail of red behind it where necessary. Notice that we (I) have been keeping track of where the cursor has ended up and what we (I) must do to get it to where it needs to be without destroying what we've created so far.

In line 85, we've introduced the PAINT instruction. Basically, all that was done was: pick an x,y coordinate that you know is

(Continued on Page 20)

Bugs, Errors, Fixes (From Page 18)

TRSDOS (26-4910)

Some users have requested an option to remove the TIME and DATE question from their TRSDOS 2.0 and 2.0a disks.

The following PATCH will omit the DATE and TIME request on boot-up. It should be noted that if this PATCH is applied, there will be no CREATION DATE or UPDATE DATE in the Directory unless you take care to type it in via the DATE command. Therefore, confusion could arise on the age of the user's programs.

1. At TRSDOS Ready, type the following patch
   PATCH SYSRES/SYS A=2367 F=73 C=CA
2. Press (ENTER)

WARNING THIS PATCH SHOULD NOT BE USED WITH ANY RADIO SHACK PRE-PACKAGED PROGRAMS. RADIO SHACK PROGRAMS USE THESE FUNCTIONS (DATE AND TIME) INTERNALLY. THIS PATCH IS ONLY FOR YOUR OWN PROGRAMS.

The following patches can be made to TRSDOS versions 2.0 and 2.0a if you need a 75 baud utility under the SETCOM utility.

These patches are designed to allow the option of selecting 75 BAUD under the SETCOM utility. This requires the deletion of 4800 BAUD as a selection since space does not allow an additional option. If circumstances require 4800 BAUD at some time or another, you simply reverse the Find and Change values to restore 4800 BAUD.

1. At TRSDOS Ready type the following patches
   PATCH SYSTEM64 R=14 B=118 F=1A47 C=D007
   and press (ENTER)
   PATCH SYSTEM/SYS R=104 B=125 F=C012 C=4B00
   and press (ENTER)
2. This completes the patches.

Color Computer (From Page 19)

inside the area to be PAINTed (60,80) & (155,103), choose the color you wish to PAINT (2 which is yellow) and the color you wish to PAINT to (4 which is red in this color set). Type in:

```
999 GOTO 999
```

and RUN the program as it is so far. What you should see is that familiar shaped house with the garage attached and a garage door without any windows. If you watch closely, you can see the roof on the house and garage being colored yellow (depending on the color adjustment on your TV set).

Press the BREAK key to get out of the program. Continue entering the rest of the program with lines 90-120:

```
90 DRAW"BL100;BU36;R96";FOR X=1 TO 6:
   DRAW"BU2;BR2;BL96;R96";NEXT X
100 DRAW"BL8;BD32;R47";FOR X=1 TO 6:
    DRAW"BU2;BR2;BL47;R47";NEXT X
110 DRAW"BD43;BL119;U22;R10;D22;BU11;BL2;D1"
120 DRAW"BD5;BL47;U12;R34;D12;L34;BR56;U12;R35;D12;L35"
```

In this series of lines, we have drawn in the door (with door knob) and the downstairs picture windows. From this point on, we will start telling the computer where to position the cursor using x,y coordinates. (It's easier than keeping track of where the cursor has ended up.)

At any point, you can stop entering lines and run the program to see what is happening. Line 999 that you entered will hold the display in a continuous loop. Simply press BREAK to continue entering program lines.

Enter the following lines to DRAW the upstairs windows, the driveway, street and sidewalk, plus windows on the garage door:

```
130 DRAW"BM63,100;D4;R6;U4;L6":
    FOR X=1 TO 4: DRAW"BR16;D4;R6;U4;L6": NEXT X
140 DRAW"BM154,135;D1;L4":
    FOR X=1 TO 11: DRAW"D1;L4": NEXT X: DRAW"L44":
    FOR X=1 TO 12: DRAW"U1;R4": NEXT X
150 DRAW"BM95,135;D1;L4":
    FOR X=1 TO 15: DRAW"D1;L4": NEXT X: DRAW"R60"
160 FOR X=1 TO 7: DRAW"D1;L4": NEXT X: DRAW"L65"
170 DRAW"BM194,135;D1;L4":
    FOR X=1 TO 23: DRAW"D1;L4": NEXT X
180 DRAW"R160"
190 DRAW"BM158,122;R12;BR5;R12"
```

Lines 200-250 add the final touches to an otherwise ordinary house:

```
200 PAINT(205,125),2,4
210 PAINT(198,133),2,4
220 PAINT(148,133),2,4
230 PAINT(152,95),2,4
240 PAINT(175,137),3,4
250 PAINT(0,0),3,4
```

Line 200 PAINTs the side of the garage (2 which is yellow). Line 210 PAINTs the front of the garage (2 again). Line 220 PAINTs the front of the house (correct, . . . yellow). Line 230 PAINTs the side of the house showing (right again, yellow). Line 240 PAINTs the sidewalk, driveway, and street (3 which is blue). Finally, line 250 PAINTs in the sky (again 3 which is blue).

If you were to change line 1 to:

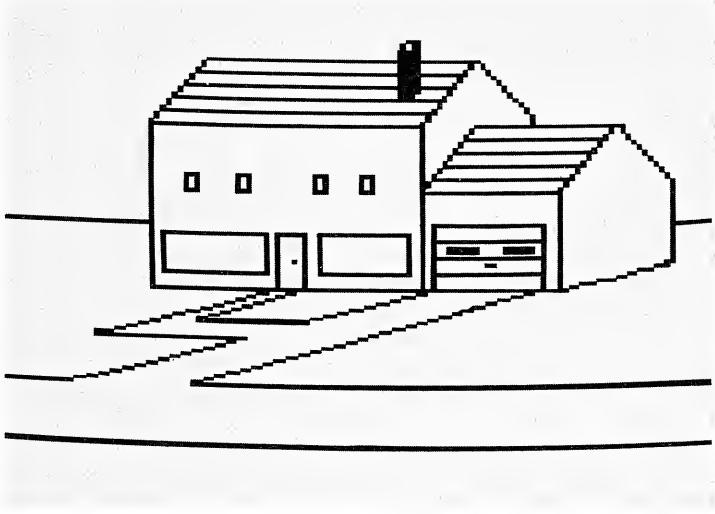
```
1 DRAW"S2"
```

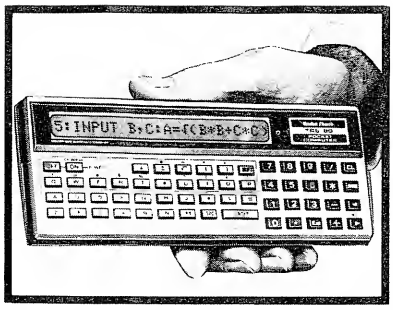
or half-scale (¾) and run the program, (you need to take out the PAINT instructions in line 85 and 200-250) you will get an idea of what Scale can do in your programming with DRAW. The entire house is reduced in size based on your Scale instruction. This drawing, however, does not accurately reduce in size due to a couple of restrictions:

1) We have specified certain starting points as x,y coordinates. When our drawing is reduced in size, those starting points may or may not be valid (in either smaller or larger sizes).

2) If you were to draw a line 25 spaces long in Scale 4, then change the scale with S2 and redraw the line, it should be 12½ spaces long. However, the computer cannot set half a space, so it rounds the value to the nearest whole number. In this case, the line would be 13 spaces long. In our program for the house, we have DRAWn lines, all of which are not evenly divisible by 2, which leads to shingles not being all on the roof, doors that are not attached where they should be, windows floating in space, and assorted other garbage scattered around on the screen. (Incidentally, if you don't delete line 85 and lines 200-250, the results might be kind of interesting.)

In any case, you should have sufficient fuel to keep the home fires burning until next month . . . So, enough for now . . .





Pocket Computer

Product Line Manager's News

This month I would like to explain some misconceptions about the Pocket Computer, correct some errors which I have found in the Owner's Manual (OM) and explore some of the capabilities and features of the Pocket Computer more fully.

Referring to page 1 of the OM under features, program capacity is stated as 1424 steps, 26 memories. Each step is equivalent to one byte and one memory is the same as 8 bytes. In addition to this there are 48 steps (bytes) of REServe program memory (see page 80). This yields a total of 1680 bytes of memory which you can use. In addition there is an 80 character input buffer which allows you to input a maximum of 80 characters on one line including the line number but not including the first ":" (colon) following the line number. This last feature has lead to some confusion. As you input a line of code, each character you type takes up one byte of the input buffer (this includes the (ENTER) key). For instance typing "PRINT" takes 5 bytes but typing "PR." as the abbreviated form only takes 3 bytes. However, if you read the OM carefully, it will tell you on page 35 by implication in the example that each statement takes only one step. To illustrate this point, try entering the following line of code:

```
950 PRINT"THIS IS A TEST OF THE":PA."INPUT CAPACITY OF
THE":PR."POCKET COMPUTER" (ENTER)
```

You will notice that the last quote mark is the 79th character and the (ENTER) key makes the 80th and acts as an instruction terminator. Now press (ENTER) and see what happens. When the display returns it looks like this:

```
950:PRINT "THIS IS A TEST OF THE":PAUSE "INPUT CAPACITY OF
THE":PRINT "POCKET COMPUTER"
```

As you use the right arrow to examine the line of code you will notice that the first colon disappears from the display and does not return unless you re-display the line of code using the up or down arrow keys. If you count the number of characters (including spaces) in the displayed line, there are a total of 87, 88 including the (ENTER)! By-the-way, if you start this experiment with nothing in your PC, you can use the MEM command to check the amount of memory steps used by subtracting from 1424. If you now move the cursor all the way to the right of the display, you will discover that you have room to add 8 more characters to this line. Try adding the following code right after the 'R' in COMPUTER:

-12345678" and press (ENTER). When the display returns go to the end of the line again and you'll notice that the last quote is missing. This is because the (ENTER) key took the place of it in memory. Another point to notice is that the line number takes up exactly as many steps as there are digits. To check this, edit the preceding line of code so that the line number is 95 instead of 950 and you'll find that you have another byte (step) of memory which you can use. This is not very clear in the example on page 35 of the OM.

Turning our attention now to page 24 and the logic functions, notice the paragraph marked with the * (asterisk). This paragraph is primarily concerned with operations performed in the direct mode from the keyboard. In the two examples, the = (equals) sign is meant to signify pressing the (ENTER) key. If you want to use these forms in an IF statement, leave off the = sign or if you want to put the result of the logical compare into a memory location you must use a form such as X=(A<0) + (A>8). This will load X with a one if A is greater than 8 or less than 0. If you want to create an XOR logical function (exclusive OR) try the following line of code:

```
100: X= (B>0) + (C>0):IF 1=X BEEP2
```

Using this format, the IF statement will be true only if B or C is greater than 0 but not if both are greater than 0 in which case X would return a 2 or if both B and C where equal to or less than 0, X would return a 0.

As further explanation of the logical expressions, the plus sign serves to add the result of the first logical compare to the result of the second. This explains the return of a 2 in the second part of Note 1 at the bottom of page 24. Looking at Note 2 then, the * simply multiplies the result of one logical compare by the result of the next logical compare. In addition, these logical compares can be strung together using the + and * signs up to the limit of nested parentheses and/or data stages. By-the-way, the – (minus) and / (divide) signs can be used to provide some interesting and in some cases useful results. However, you should be aware of a possible divide-by-zero error if you try the latter form.

Considering the examples then, if for instance we said X=(5+8) > (3*4), after pressing (ENTER), X would have a value of 1. This will work the same way in a line of BASIC code and we could then use a statement such as IF X GOTO 100 and in this case the program would go to line 100. If we were to change the 8 to a 7 then X would return a value of 0 and the program would go to the next line number to execute. If you would like to prove this to yourself, try the following program:

```
100: X=(X+11) <> (3*4) :PAUSE X
110: IF X BEEP X+1 :GOTO 100
120: BEEP X+1 :GOTO 100
```

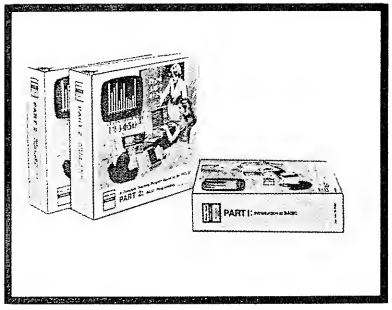
When you run this program you will see X change (on the right of the display, see below for more on this) from 1 to 0 and back while alternately hearing two beeps then one beep.

Referring now to page 46 of your OM, let's have a look at specifying variables in the fixed and flexible memories. First of all, there is an error in the paragraph marked by the third black dot. The sentence beginning "And E, B(5) . . ." etc. is completely wrong and should be marked out and the paragraph is now correct. This correction should make clearer the fact that referencing flexible memories can only be done in the form A(n) or A$(n) where 'n' is a fixed memory (A thru Z), a numeric expression (ie 3*4) or an indirect designation such as A(C) or A(3). Examination of the chart on page 47 should help to clarify this concept. An easy way to determine the highest flexible memory which can be used for data is to use the MEM command. Simply add the number of memories remaining to 26 to determine this highest address. A quick note, in the example under Flexible memory, A$(19*2) specifies flexible memory A$(38) and not A$(28).

Turning to the bottom of page 52, the last sentence, the word 'appear' should be 'disappear.' Over on page 53, the section under the Note:, the third sentence; the text refers to 'indexing the (SHIFT) and (") keys' . . . this means that if you are inputting a character variable such as ACCT#12, if you accidentally hit the " (quote) instead of the # (pound) sign, all input after the " (quote) will be ignored.

Referring now to page 55, the first paragraph, second sentence; this sentence should read "The left section displays . . . and the right section another set." The right section displays the Expression only if it is in General Form (1). If you have used the PRINT statement at all, you have probably noticed this by now but I thought it was worth pointing out anyway.

(Continued on Page 13)



Education

Educational Products News

Using VARPTR to Compress Graphic Strings

Dennis F. Tanner

In educational applications, TRS-80 graphics can be effective in helping to convey, clarify, and emphasize the meaning of other information presented by the computer.

Graphics may be produced with the TRS-80 Models I and III in a variety of ways: the SET and RESET statements, POKEing character values into the video RAM, and building them using character codes. One more method is determining a string variable's location in memory using the VARPTR statement, and POKEing character values directly in the variable storage area. This last method will be examined in depth in this article.

For comparison's sake, let's first examine one of the other methods, building string variables using graphics codes. A graphics shape resembling a capital "H" may be created using this BASIC routine:

```
10 C$= CHR$(157)+ CHR$(140)+ CHR$(174)
20 PRINT C$
```

This works, but it uses space inefficiently. On the right side of the assignment statement in line 10, twenty bytes of memory were used to build a three-character string!

CHR$	used 3 times =	3 bytes
"(" and ")"	used 6 times =	6 bytes
"+"	used 2 times =	2 bytes
Digits	used 9 times =	9 bytes
	Total =	20 bytes

String characters can be built much more efficiently by working with the variable more directly. Using the VARPTR statement, we can determine much about how variables are stored. From pages 8/13-8/14 of the Model I Level II BASIC Reference Manual, and pages 8/9-8/10 of the Model III BASIC Reference Manual, we find that this information is available to the BASIC programmer regarding the storage of string variables:

VARPTR(string variable) — returns a memory location, K.

PEEK(K) — returns the length of the string (variable).

PEEK(K + 1) — returns the least significant byte of the starting address where the variable is stored.

PEEK(K + 2) — returns the most significant byte of the starting address.

PEEK(K − 2) and PEEK(K − 1) — when combined, return the ASCII code for the variable name.

PEEK(K − 3) — returns a descriptor code identifying the variable type (the value 03 stands for string variables).

Using this information, we can write the following program to give us information about a string variable:

```
60000 C$="␣␣␣"
60010 K= VARPTR(C$): IF K>32767 THEN K= K- 65536
60020 PRINT "VARPTR =" K
60030 PRINT "LENGTH OF STRING =" PEEK(K)
60040 PRINT "LSB OF STARTING ADDRESS =" PEEK(K+1)
60050 PRINT "MSB OF STARTING ADDRESS =" PEEK(K+2)
60060 S= PEEK(K+1) + PEEK(K+2) * 256: IF S>32767 THEN
      S= S- 65536
60070 PRINT "STARTING ADDRESS = " PEEK(K+1) "+"
      PEEK(K+2) "* 256 =" S
60080 PRINT "CHARACTER CODE OF VARIABLE NAME ="
      PEEK(K-2); PEEK(K-1)
60090 PRINT "TYPE OF VARIABLE =" PEEK(K-3)
60100 PRINT "ASCII CODES OF VARIABLE =";: FOR J=0 TO
      PEEK(K)-1: M=S+J: IF M>32767 THEN M=M-65536
60110 PRINT PEEK(M);: NEXT J: PRINT
60120 PRINT "C$ =": PRINT C$
```

Now that we know exactly where each byte of the string variable is stored, we can POKE new values into those locations by adding the following lines to our program:

```
60130 FOR I=1 TO PEEK(K)
60140 PRINT "VALUE FOR BYTE" I;: INPUT V
60150 M= S+ I- 1: IF M>32767 THEN M=M-65536
60160 POKE M, V
60170 NEXT I
60180 GOTO 60000
```

NOTES:

1) This routine is designed to be appended to an existing program, as described later. After you have assigned graphics codes to all desired variables, the entire routine can be deleted.

2) In this routine, the slashed lower case b ("␣") is used to represent a space that is produced by pressing the space bar.

If you run the program now, and input the values 157, 140, and 174 (the same values we used in our first program), the string variable C$ will contain the same characters as it did in the first program.

When you list the program again, you will see something that you might not have expected. On the Model I, Line 60000 now reads:

```
60000 C$= "EDITLETSYSTEM"
```

This is because you have POKEd the values for the individual parts of the character into the variable C$, and the computer interpreted them as reserved words, as listed on page E/1 of the Level II Manual. On a Model III, the listing of the line 60000 would contain the graphics string itself.

Note that in lines 60010, 60060, 60100, and 60150 we compared the value of the memory location to 32767. This is because the memory locations "wrap around" to the negative locations. That is, for purposes of PEEKing and POKEing, these locations are in ascending order: 1, 2, ..., 32765, 32766, 32767, −32768, −32767, −32766, ... −3, −2, −1. If the address of the location reaches 32769, for example, the program subtracts 65536 to compute the appropriate value, −32767.

We have now accomplished our goal. Line 60000 now assigns the graphics string with five bytes on the right side of the assignment statement (compared to the twenty we used before). If you don't believe it, PRINT LEN(C$) to be sure. The three bytes of the character, plus two sets of quotation marks, makes the five bytes.

To use this routine to make your programs more efficient, append this program onto the end of your regular program. At the point in the program where you want to assign the characters to the string variable, make an assignment statement of the proper length, using all spaces. Here's an example using MA$ as the variable name:

```
100 MA$= "␣␣␣␣␣␣␣␣␣"
101 GOTO 60010
.
.
(remainder of main program)
.
.
```

(Continued on Page 23)

Education (From Page 22)

Line 60000 may be DELETEd from the program since it will not be used. Line 60010 should be changed to read:

```
60010 K= VARPTR(MA$)
```

and line 60120 should read as follows:

```
60120 PRINT "MA$ = ": PRINT MA$
```

Since line 60000 has been DELETEd, line 60180 must be changed to read:

```
60180 END
```

Then execute a RUN 100 to begin the program. Simply input the ASCII values you want to use for the variable, and they will be assigned to the variable. Note that the ASCII values for letters, numbers, and other characters or codes may be used, too.

After all your graphics variables have been assigned, the routine at the end of the program (60010-60180) may be DELETEd from the program if you will not need it again.

(Don't forget to DELETE line 101, too!)

In closing, let me leave you with this coded message:

```
MA$= "¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥"
```

Use these values for data: 66, 89, 69, 33, 128, 132, 132, 132, 132, 128, 68, 69, 78, 78, 73, 83.

Model I/III Program Compatibility

Here is a summary of Radio Shack software, and how it does (or does not) work with a Model III:

The following programs use one or more of the characters:

CHR$(91) (Mod I—↑, Mod III—[)
CHR$(92) (Mod I—←, Mod III—\)
CHR$(93) (Mod I—↓, Mod III—])
CHR$(94) (Mod I—→, Mod III—^):

Budget Management 26-1603
Casino Games 26-1806
Cassette Portfolio 26-1506
Games Pack I (Space Taxi) 26-1805
Invasion Force 26-1906
K-8 Math 26-1715
Level I BASIC Instruction Course 26-2003
Level II BASIC Instruction Course—Part I 26-2005
Level II BASIC Instruction Course—Part II 26-2006
Micro Music 26-1902

You should be aware of this and recognize which characters are used on the Model III video to represent the arrow of the Model I. Hint: I have trouble remembering which character is which arrow, so I taped the symbols to the front of the arrow keys on the keyboard. Now when I see an "arrow character [\ ] ^" I can look at the keyboard and match the character with the proper arrow.

These differences are cosmetic only and do not affect the execution of the program.

The following is a list of Model I programs which need minor corrections to allow them to function correctly in a Model III:

Advanced Statistical Analysis 26-1705
Budget Management 26-1603
Casino Games 26-1806
Cassette Mailing List 26-1503
Cassette Portfolio 26-1506
Dancing Demon 26-1911
Level II BASIC Instruction Course—Part II 26-2006
Real Estate Volume III 26-1573
Real Estate Volume IV 26-1574

The corrections to make these programs run on a Model III are included with the Model III. However, if you do not wish to make the changes yourself, ask your Radio Shack store manager to return the tape for exchange with the Model III version.

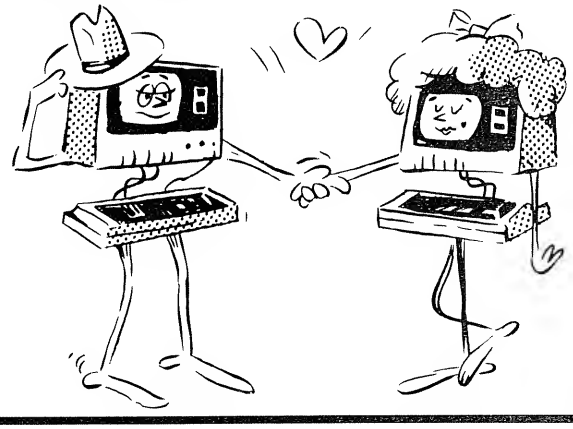
The following programs cannot easily be changed to Model III versions. As we replenish our stock of these programs (and make the appropriate changes), we will include tapes or disks (as appropriate) for both Model I and III in the package. If the program is a disk program, it may be "convertible" rather than having disks for both Model I and Model III. We have gone back and reworked a lot of our warehouse stock, so you will have the Model III software as soon as possible. If you happen to get a copy of this software which does not contain the Model III version, ask your store manager to order the proper version for you. The 700-XXXX number shown is the Model III exchange version and will be provided free of charge. If the package does not have a 700 number, or if it has the † symbol, we have not released the Model III exchange version yet, but one is coming.

Business Mailing List 26-1558 (700-2212)
K-8 Math 26-1715 †
Mailgram™ 26-1564 (700-2004)
Micro Movie 26-1903 †
Microfiles 26-1565 (700-2012)
Profile 26-1562 (700-2010)
Scripsit™—Disk 26-1563 (700-2006)
Scripsit™—Tape 26-1505 (700-2005)
Standard and Poor's Stockpak™ System 26-1507 †
Versafile 26-1604 (700-2008)

The following software will NOT run on a Model III, and is NOT available as an exchange item. Do not buy this software for a Model III computer! Most of the items in this list (or a package of a similar nature) will be (or are) available for the Model III (these are marked with an *). When available, the Model III version may be packed with the existing Model I program, or we may issue it as a new stock number for Model III only. We will keep you informed about these programs through the Newsletter, and our local stores.

Disk BASIC Instruction Course 26-2007 *
Editor Assembler—Disk 26-2202 *
Editor Assembler—Tape 26-2002 *
Micro Chess 26-1901 *
Renumber 26-2004
RS-232 Communications Pack 26-1146 *
RSTERM 26-1147 *
Space Warp 26-1912 *
T-Bug 26-2001 *
Tiny Pascal 26-2009 *
TRS-80 FORTRAN 26-2201 *
VisiCalc 26-1566 is available for Model III.

All programs not mentioned above do not require modifications to work on a Model III.



Radio Shack®
MICROCOMPUTER NEWS
P.O. BOX 2910
FORT WORTH, TX 76101

BULK RATE
U.S. POSTAGE
PAID
Radio Shack
A Div. of Tandy Corp.

ADDRESS CHANGE
☐ Remove from List
☐ Change as shown
Please detach address label and mail to address shown above.

IF UNDELIVERABLE DO NOT RETURN

///

Line Draw

Eddie Pont, Southfield MI

I read your newsletter and enjoy its programs very much. The only thing missing in a lot of these programs is a chance for the user to create.

This program for Level II Model I's, or Model III's with Model III BASIC will start a graphics line at 0,0 and move it to the right. The user has the ability (via the INKEY$ statement) to control the direction of the line. It is very relaxing and can be fun for hours on end.

```
1 REM LINE DRAW
3 REM BY: EDDIE PONT*** JAN, '81
5 REM "U"=UP: "D"=DOWN
7 REM "R"=RIGHT: "L"=LEFT
10 CLS: ON ERROR GOTO 150
20 B$="R"
30 SET(X,Y): FOR T=1 TO 100:NEXT
40 A$=INKEY$: IF LEN(A$)>0 GOTO 100
50 IF B$="U" THEN Y=Y-1: GOTO 30
60 IF B$="D" THEN Y=Y+1: GOTO 30
70 IF B$="R" THEN X=X+1: GOTO 30
80 IF B$="L" THEN X=X-1: GOTO 30
90 A$="": GOTO 30
100 B$=A$: A$=""
110 GOTO 30
150 IF ERR/2+1=5 GOTO 160
    ELSE RESUME 40
160 PRINT@ 896,"OUT OF RANGE";
170 A$=INKEY$: IF LEN(A$)=0
    GOTO 170
180 PRINT@ 896,"
190 B$=A$: A$="": RESUME 50
```

Mr. Pont notes that if you get the "OUT OF RANGE" error you must move in the opposite direction.

System

Andrew Koledin

In the Jan. 1981 issue, you gave us a short program to find the name of a "SYSTEM" tape on Model I. While that program does work, I have a much shorter program that you may pass on to your readers. It also finds the name and is very simple. You may print it in your Newsletter. (Just spell my name right!)

```
10 CLEAR 300
20 INPUT #-1, A$
30 PRINT MID$(A$,2,6)
```

Editor's note: This works for both Model I and Model III System tapes.

Burma Shave

Here is a quick little program from Edward L. Wilkening II of North Hollywood, CA which may bring back some memories for many of you:

```
10 CLS: PRINT CHR$(23): A=14
20 PRINT@ 980, "EINI MINI":
   GOSUB 100
30 PRINT@ 980, "MINI MOE":
   GOSUB 100
40 PRINT@ 974, "COMPUTERS NEVER":
   GOSUB 100
50 PRINT@ 980, "SAY HELLO":
   GOSUB 100
60 PRINT@ 978, "BURMA SHAVE": A=7:
   GOSBU 100
70 FOR Z= 1 TO 300: NEXT Z
80 CLS: FOR Q= 1 TO 300: NEXT Q
90 GOTO 10
100 FOR N=1 TO A
110 FOR I=1 TO 20: NEXT I
120 PRINT
130 NEXT N
140 RETURN
150 END
```

More Computer Clubs

Western New York TRS-80 Users Group
c/o Nick Rekito
6040 Baer Road
Sanborn, N.Y. 14132

Robert Stuart Computer Club
Robert Stuart Junior High School
Caswell Ave. W.
Twin Falls, Idaho 83301

Reno Computer Club
P.O. Box 11874
Reno, NV 89510
702/322-7988

COMPUTER SERVICES ADDRESS AND PHONE NUMBERS
8AM to 7PM Central Time

Computer Services
900 Two Tandy Center
Fort Worth, Texas 76102

Model I/III Business Software
Outside Texas 1-800-433-5641
In Texas 1-800-772-5973*

Model II Business Software
Outside Texas 1-800-433-5640
In Texas 1-800-772-5972*

All Other Calls
Outside Texas 1-800-433-1679
In Texas 1-800-772-5914

Switchboard — 1-817-390-3583
*Available May 20, 1981

TRS-80 Microcomputer News is published monthly by Radio Shack, a division of Tandy Corporation. A single one year subscription is available free to purchasers of new TRS-80 Microcomputer systems with addresses in the United States, Puerto Rico, and APO or FPO addresses. Subscriptions to other addresses are not available. The subscription rate for renewals and other interested persons is twelve dollars ($12.00) per year, check or money order in U.S. funds. All correspondence related to subscriptions should be sent to: Microcomputer News, P.O. Box 2910, Fort Worth, Texas 76101.

Back issues of Microcomputer News are not available.

The TRS-80 Newsletter welcomes the receipt of computer programs, or other material which you would like to make available to users of TRS-80 Microcomputer systems. In order for us to reprint your submission, you must specifically request that your material be considered for reprinting in the newsletter and provide no notice that you retain copyrights or other exclusive rights in the material. This assures that our readers may be permitted to recopy and use your material without creating any legal hassles.